ESCUELA SUPERIOR DE INGENIERÍA

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

VocabTrainer A1:
Sistema móvil para la gestión de actividades de aprendizaje ubicuo de lengua extranjera

Manuel Jesús Pérez Zurera

23 de abril de 2014

ESCUELA SUPERIOR DE INGENIERÍA

INGENIERÍA TÉCNICA EN INFORMÁTICA DE SISTEMAS

VocabTrainer A1:
Sistema móvil para la gestión de actividades de aprendizaje ubicuo de lengua extranjera

- Autor del proyecto: MANUEL JESÚS PÉREZ ZURERA
- Director del proyecto: MANUEL PALOMO DUARTE
- Departamento: Ingeniería Informática
- Directora del proyecto: ANKE BERNS
- Departamento: Filología Francesa e Inglesa

Cádiz, 23 de abril de 2014
Fdo: Manuel Jesús Pérez Zurera

Agradecimientos

A Maria Josefa Fernandez de los Reyes, por que sin ella yo no hubiera llegado hasta aquí.

A Maria del Carmen Zurera Fernandez y a Ramon Jimenez Saltares, sobran los motivos para agradeceros.

A Manuel Palomo Duarte por ofrecerme su ayuda en el proyecto y guiarme por el desarrollo del mismo.

A Cristina Bueno Delgado por ser la persona que más interés ha mostrado en el proyecto y más horas de dedicación sin tener por qué.

A Anke Berns por su gran esfuerzo y dedicación en el proyecto codo con codo.

A mis amigos, quienes después de tantos años saben perdonar y comprender para que una amistad no decaiga.

A Pilar Romero Sevilla, Ignacio Calleja Olmedo, Andrea Márquez Calderón, Patricia Atienza Cuevas, Karina Andrades Silva, Alexia Zilliox, Juan Antonio Sánchez Ferrera y David Romero Santos, Hanna Kiemle, Kerstin Walentowski, Hannah, Lisa Steigerwald, Jana Borghoff, Tim Bauer, Hannah Koch, por colaborar en la elaboración del material audiovisual de la aplicación, gracias de corazón

A Elihú Salcedo Ruíz, por administrar y configurar todo el servidor de Openfire, gracias por la ayuda.

A todos los compañeros del instituto IES Poeta García Gutierrez, Mar Dodero, Cristina Ortega y César Juan Roldán, gracias por ayudarnos en las pruebas y por ser tan amables con nosotros.

Licencia

Resumen

Hoy en día las tecnologías avanzan de manera arrolladora. Nuestro cometido es dotar de significado y valor a esas tecnologías. Se debe valorar el esfuerzo para intentar adaptar esos avances en nuestra vida cotidiana, en la enseñanza, en las relaciones sociales, etc. Este proyecto es fruto del intento una vez más de querer acercar los últimos descubrimientos a la vida del estudiante.

La aplicación desarrollada en este PFC, hará uso de diferentes herramientas para acercar el conocimiento al alumno, y para poder proporcionarle una fuente precisa y contrastada de vocabulario. Gracias al uso de imágenes de objetos, voces de personas nativas del idioma, y de la interacción con otros alumnos, haremos un aprendizaje más interactivo que servirá como complemento perfecto a la enseñanza tradicional.

La aplicación se compone de diferentes partes bien definidas:

- Actividades de relación de objetos: el alumno deberá elegir el objeto correcto y se le premiará con la pronunciación exacta del objeto.
- Actividades de aportación escrita: el alumno deberá rellenar con texto los huecos indicados.
- Actividad en grupo mediante chat: tres alumnos deberán realizar una actividad conjunta en la que deberán superar una prueba en tiempo real, mediante la interacción directa con códigos QR.

Todo ello ha supuesto la investigación de tres tecnologías y poder hacerlas funcionar de manera correcta: Android (para poder dotar de movilidad a las pruebas), Códigos QR (para poder proporcionar contacto físico entre objeto y palabra) y servidor Chat (indispensable a la hora de querer hacerlo social).

Índice general

Índice de figuras

Parte I

Prolegómenos

Capítulo 1

Introducción

1.1. Motivación

Desde siempre he estado interesado en el mundo de los videojuegos y de las tecnologías, tanto como parte de ocio como parte a desarrollar por mi mismo. La motivación de este proyecto surge de querer proporcionar a la sociedad una herramienta divertida y agradable para poder repasar conocimientos de manera fácil.

Hoy en día todos llevamos dispositivos móviles con los que usamos diferentes aplicaciones que nos facilitan la vida un poco. Si utilizamos el móvil, la tablet, el portátil... para jugar y para divertirnos, ¿por que no aprovechar sus ventajas también en la enseñanza?. Aprender un idioma siempre es algo que conlleva un gran trabajo. Desde el esfuerzo de tener el primer contacto con el idioma al esfuerzo repetitivo de tener que aprender el vocabulario a veces de memoria hasta que se nos quede grabado.

La idea con la que surgió el proyecto fue la de hacer que el alumno pueda entrenar sus conocimientos ya adquiridos de manera directa y como si de un videojuego se tratase. Ese enfoque es el que nos ha hecho trabajar pensando siempre en que como alumnos que hemos sido y somos, tengamos otras maneras más amenas de mejorar nuestros conocimientos y tener una alternativa para el autobús, para el tren, para una tarde con un café en un bar... y poder poner en práctica nuestros conocimientos.

¿Por qué elegir Android? es una tecnología que me ha parecido que ha llegado a conectar rápidamente con la sociedad, y de la que podemos sacar mucho partido si queremos llegar de la manera más accesible al alumno. ¿Cuantas veces usamos el móvil a diario? ¿Cuantas de ellas por diversión? La cuestión es que estamos muy acostumbrados a usarlos en nuestro día a día y desarrollar una aplicación como ésta en tal plataforma ha hecho que tenga unos requisitos que la mayoría de usuarios llega a cumplir.

1.2. Objetivo

El proyecto tiene distintos objetivos, ya que no abarca uno solo es importante darle importancia a todos y cada uno de sus apartados.

- Proporcionar al estudiante una herramienta ubicua de aprendizaje a fin de facilitar a los alumnos el aprendizaje de diferentes aspectos lingüisticos y temáticos que se basan en el Nivel A1 del Marco Común Europeo de referencia para las lenguas (MCERL). [12] [14] [13] [15]
- Capacidad de autoevaluación: con este 'trainer' el alumno será capaz de poner a prueba sus conocimientos y además podrá comprobar en que se ha equivocado, cuando, y cual es la respuesta correcta. En cualquier caso (equivocarse o acertar) el alumno también tendrá a su disposición el tiempo en que ha tardado en resolver el ejercicio. [17]
- Aprendizaje cooperativo: una vez desbloqueadas todas las nueve actividades los jugadores deberán pasar una última prueba para demostrar los conocimientos lingüisticos aprendidos anteriormente mediante los minijuegos: un juego cooperativo que se basa en una gymkana que tiene como objetivo resolver un crimen ficticio. [16] [19]
- Relación de objetos: los jugadores deberán identificar una serie de objetos que están colocados físicamente en el lugar donde se realiza la gymkana. [11]
- Evaluación docente: una meta a cumplir era que el propio profesor pudiera evaluar el uso de la lengua alemana durante el juego colaborativo por parte del alumno teniendo acceso al registro de la conversación completa de la actividad. [16] [18]

En definitiva podemos decir que el objetivo de la aplicación es proporcionar al usuario un conjunto de herramientas rápidas, fáciles y sobre todo interactivas para poder desarrollar sus conocimientos y dotarles de capacidad de autoevaluación.

1.3. Estructura del documento

En esta sección vamos a comentar el contenido de cada sección de la memoria.

- **Estado del arte** En esta sección, se tratará de comentar los detalles de otros sistemas parecidos al nuestro, también de los entornos y las herramientas para la investigación propia del proyecto.
- **Planificación** La planificación del proyecto, junto a las iteraciones que ha seguido a la hora de ser desarrollado, serán detalladas en esta sección junto a un diagrama ilustrativo.
- **Análisis** En esta sección, trataremos del propio análisis de la aplicación, sus casos de uso y las clases que lo componen.

- **Diseño** En esta otra sección, se hablará del diseño de VocabTrainer A1, hablaremos sobre el software utilizado para crear contenido, de cómo están pensadas las actividades, etc.
- **Implementación** Aquí comentaremos los problemas que nos encontramos a la hora de pasar a la práctica el diseño del sistema, que soluciones hallamos, que medidas tuvimos que tomar, y hasta una estadística de la propia implementación de la app.
- **Pruebas** En este apartado comentaremos en profundidad las pruebas realizadas a la app a nivel técnico, y a nivel de uso y educativo con pruebas con los propios alumnos.
- **Apéndice** Siendo la sección apéndice, trataremos de temas como las posibles ampliaciones que podría tener el sistema, o de como se muestran los registros de las conversaciones.
- **Manual de usuario** En manual de usuario encontraremos ciertas directrices para el manejo e instalación del servidor Openfire, recurso indispensable para proporcionar soporte a las partidas de Catch me, if you can!.
- **Manual de la aplicación** En esta sección encontraremos pasos detallados de como instalar la aplicación, que requisitos tiene, su configuración, su uso y hasta su desinstalación.
- **Conclusiones** En el penúltimo punto, expongo mis conclusiones del proyecto, los aspectos que mas me han gustado y los que menos, que podría mejorarse de querer abarcar un proyecto más ambicioso, etc.
- **Bibliografía** En este apartado enlazaremos a artículos y publicaciones que han servido a la hora de desarrollar e idear el proyecto, así como servir de inspiración y de base para el mismo.

Capítulo 2

Estado del arte

2.1. Comparación con otros sistemas

En esta sección compararemos nuestro proyecto con otros sistemas ya existentes. Hay que reconocer que nuestra aplicación tiene dos partes bien definidas. Una enfocada al modo individual en el que se pondrán a prueba los conocimientos sobre el vocabulario del alumno (y del manejo de éste en frases en presente) y otra parte colaborativa en el que hará falta un poco más de ingenio para resolver el reto que se propone.

Por ello, necesitaremos compararlo cada faceta con un sistema diferente ya que a priori no hay sistemas conocidos que intenten mezclar las dos facetas en un mismo paquete. Lo que si hemos encontrado es comparaciones con otros sistemas que posean **gamificación**. Gamificación es un término nuevo que indica la cualidad de convertir una tarea que en principio es tediosa y repetitiva, en una que proporcione un reto interesante y se le proporcionen premios al que la realice mediante el uso de recompensas y/o trofeos ficticios.
Hemos encontrado similitudes con los siguientes sistemas.

2.1.1. Brain training

Brain training es un videojuego muy famoso lanzado en 2006 para la consola Nintendo DS, que tuvo gran aceptación en todo el mundo. Esto es debido a varios factores, uno de ellos es la facilidad para introducir datos al juego.

Antes del lanzamiento de consolas como Nintendo DS, la introducción de datos en los videojuegos se hacía en su mayoría mediante un mando con teclas, esto precisaba de que el jugador conociese de antemano el uso de un mando y conocerse bien la ubicación de los botones en el mando. Esto suponía un lastre a la hora de querer introducir frases complejas o de elegir una secuencia de números ya que se debía disponer de un teclado por pantalla o una solución igual de compleja.

La facilidad de uso que aportó la consola fue que la introducción de datos se hacía mediante

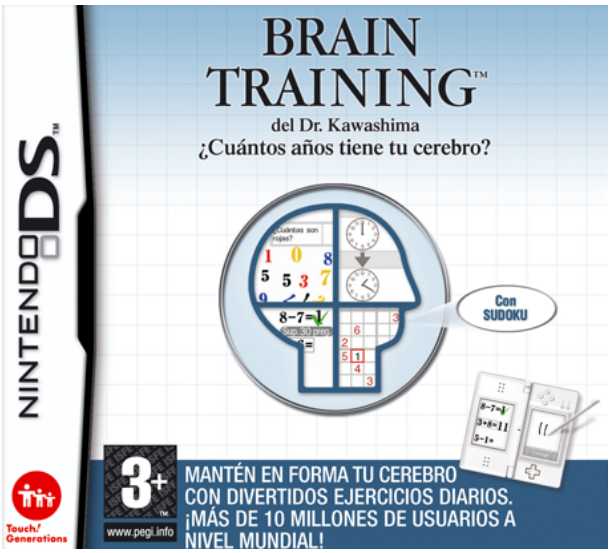


Figura 2.1: Portada del videojuego Brain Training

un lápiz, igual que ciertos aparatos electrónicos que empezaban a coger auge por esa época. El juego que estamos citando, proponía una serie de retos matemáticos para el jugador, mediante los cuales prometía mejorar la habilidad matemática del propio jugador (al menos esa era la promesa del juego).

¿Por qué llego a tanta gente? Porque no precisaba de conocer los mandos de ninguna consola ni entender de videojuegos, simplemente el jugador debía entender una pregunta y responder con un lápiz de la misma manera que lo haría sobre un papel. Esto produjo que el juego llegara a gente de cualquier edad, y no sólo a los jóvenes conocedores de los videojuegos. Por lo tanto el impacto fue muy grande debido a su método de jugabilidad.

En comparación con Brain training, nuestro proyecto toma una idea parecida. Nosotros tampoco hemos tomado la vía clásica de los videojuegos de pulsar difíciles combinaciones de botones si no que se le proporciona una pregunta al jugador y la respuesta está a un click con el dedo. Elegir una respuesta puede ser tan natural como tocar la foto de un hotel o de una piscina, si es que pensamos que esa es la respuesta.

Yendo un paso más allá, tenemos que podemos introducir frases no muy complejas por teclado. Este aspecto podría ser más complejo para el jugador que no conozca el uso de un Smartphone, pero cada día se ven más y más personas que son capaces de utilizar el móvil para muchos aspectos de su vida cotidiana como enviar un correo electrónico.

Otra de las grandes bazas con las que jugaba Brain training, era que mostraba estadísticas de tus avances con el propio juego. Esto era un aspecto muy importante ya que la recompensa (parte importantísima de la gamificación) era precisamente mostrar al usuario lo bien que lo estaba haciendo comparado con retos anteriores.

Nosotros hemos hecho algo parecido aunque quizá no tan elaborado. Más que mostrar los avances del jugador, mostramos un número (en la mayoría de los apartados tres, en algunos sólo dos) de aprobados por actividad. Si has sacado una cierta nota, esta puntuación aparecerá con un tick verde señalando que esa actividad que se hizo en tal día con tal puntuación y se resolvió con tal tiempo está aprobada. En cambio si no fue superada aparecerá con un aspa roja indicando que se debe repetir para poder aprobar y seguir adelante con el juego.

Por si fuera poco y para rizar más el rizo, Brain training hacía una suposición basado en tus puntuaciones de la edad de tu cerebro. Este aspecto no nos ha parecido atractivo ya que, aunque supuso una manera de 'enganchar' a los jugadores para que mejoraran su edad, no hay estudios que avalen esta cualidad. Más bien sucede al contrario, según estudios de la revista 'Nature' en colaboración con la BBC, no existe mejora directa en del jugador en sus capacidades cognitivas, aunque si destacan la mejora en la práctica del jugador en el uso del conocimiento que trae el juego. Es decir, que por mucho que jugase alguien a Brain training no iba a mejorar su capacidad para entender las matemáticas, aunque si podía repasar conocimientos de ésta que tenía algo olvidados.

Es por ello, por lo que tenemos un enfoque un tanto distinto con el proyecto, no se trata de aprender Alemán, Inglés o el idioma que tenga implementado, es un entrenador (trainer) para poner en forma tu vocabulario del idioma, y para saber usarlo en su contexto o en pruebas de la vida cotidiana. Nuestro proyecto no promete sustituir a ningún método de enseñanza directo, pretende ayudar y servir de entrenamiento al usuario para que se le haga más fácil el aprendizaje del idioma.

Por último, un punto muy positivo a destacar de ambos sistemas, es que ambos son para dispositivos portátiles, por lo que brindan una facilidad de acceso al juego muy amplia. Brain training no hubiera tenido la aceptación que tuvo si sólo pudiese haber sido jugado en una Playstation 2, ya que supondría que no se podría jugar en el metro, en la sala de espera del dentista... en definitiva en cualquier lugar dónde se quiera jugar. Nosotros vamos un paso más allá y tenemos que no se precisa de ninguna consola para poder entrenar tus conocimientos si no que podrás llevar el trainer en el mismo sitio dónde llevas el Smartphone que tanto se utiliza hoy día, por lo que aseguramos la accesibilidad a la aplicación en casi cualquier momento del día.

2.1.2. SuperHub

En los comienzos de nuestro proyecto, quisimos enfocarnos a un modo de juego que fuera por geolocalización por GPS. Queríamos integrar el aprendizaje con la vida cotidiana, a los lugares que conocemos. SuperHub es un buen ejemplo de ello. Siendo un proyecto de software libre en el que se han invertido unos 10M de euros y unos 36 meses de desarrollo, captó nuestra atención como inspiración para el proyecto. A finales del año pasado se hicieron pruebas de concepto de tal proyecto en Barcelona por lo que el proyecto está empezando a funcionar todavía.

Figura 2.2: Icono del curso de idiomas Voxy

La idea inicial de Superhub era la de proponer minijuegos para promover el uso de los distintos métodos de transporte urbano. Tenía como reto tener que llegar a ciertos puntos del mapa de tu ciudad a los que tenías que llegar y marcar como completado. Si no podías cumplir tu *misión* podías recaer la tarea en otra persona y ésta podía colaborar contigo para cumplirla. De esta manera ibas conociendo las distintas maneras de desplazarte por tu ciudad de la mejor manera posible, gastando pocos recursos energéticos.

Aunque el proyecto ha evolucionado y ahora es capaz de predecir la congestión del tráfico y recomendarte la mejor manera para desplazarte de un punto a otro, la idea inicial era una gamificación del desplazamiento urbano del ciudadano. Al principio pensamos en tomar la misma vía para el desarrollo colaborativo. Pensamos en poner como reto tener que ir a la playa, o tener que acercarte al parque para poder ganar la partida.

Tuvimos que desechar esas ideas ya que teníamos bastantes inconvenientes. Uno no muy importante era que es dependiente del lugar y por tanto, no es aplicable a todos los lugares universitarios. Otro y más importante, era la cantidad de problemas que podía suponer dar una actividad universitaria por la calle fuera de las aulas. Por lo tanto parecía que nuestra única opción era hacerlo dentro de la propia facultad, dónde la cobertura GPS era muy baja o nula. Por estos motivos finalmente optamos por un enfoque no geográfico y más por un enfoque de objetos con los códigos, ya que nos daba más maniobrabilidad con los retos dentro de un edificio.

2.1.3. Voxy

Aunque no nos inspiramos en este proyecto, se puede decir que está relacionado con el aprendizaje del idioma en las nuevas tecnologías. Voxy propone como método de aprendizaje el uso de vocabulario personalizado y actualizado directamente desde tu dispositivo móvil o ordenador de casa.

Aunque el propósito de Voxy es parecido al de nuestro proyecto, su manera de abarcar el problema es mucho más amplia y compleja. Desde la página web podemos ver cómo funciona su

método de aprendizaje. Cada día te envían una noticia real relacionada con tu nivel de inglés y tu zona geográfica, o nos envían una lección relacionada con nosotros. También podemos aprender con la música, podremos elegir entre las que tienen gratuitas para poder leer la letra (y escuchar la canción) o comprar de las que haya en venta en la tienda.

Con esto pretenden lo mismo que nosotros hemos pretendido con nuestro proyecto, dar un vocabulario cotidiano y evitando expresiones sacadas de los libros o de antiguas lecciones. Aunque eso sí nuestra aplicación no se actualiza para descargar nuevas lecciones sí que contiene vocabulario cotidiano para el estudiante.

También prometen una tecnología de reconocimiento de voz, en la que podrás repasar tu pronunciación del idioma con tu dispositivo. Otra característica a destacar es la posibilidad de tomar una foto y que el sistema reconozca que objeto es y nos diga su nombre en el idioma que estamos aprendiendo. Todas estas características están disponibles desde la aplicación para iOS y Android, y aunque es gratuita sólo nos ofertan 7 días de prueba. Más tarde habrá que elegir entre las diferentes subscripciones que proponen.

Aparte, también permite la comunicación con profesores nativos que te ayudaran a superar el curso al que estés subscrito, convirtiendo toda la experiencia en una especie de curso personalizado en tu móvil. En cambio, nuestra aplicación no parte de la idea de hacer un curso personalizado con palabras sacadas de noticias o de lecciones actuales. Como ya hemos comentado es un trainer que sirve para entrenar un vocabulario estático (aunque se podría ampliar con facilidad).

La gran diferencia radica en que Voxy sería un curso que promete sustituir a la enseñanza tradicional para la gente que no disponga de tanto tiempo como para asistir a clases y seguir recibiendo un vocabulario contrastado y de confianza, mientras que nuestro proyecto se limita a servir de herramienta de ayuda al estudiante que esté cursando los estudios de un idioma. Debido a esta diferencia, Voxy como aplicación necesitará de una subscripción de pago para el curso para funcionar, y VocabTrainer A1 es gratuita y libre para cualquier estudiante.

2.1.4. OpenSim

OpenSim se ha utilizado para el desarrollo de un proyecto de aprendizaje por parte de Raúl Gómez Sanchez como proyecto final de carrera en colaboración con Anke Berns, Daniel Molina Cabrera, Iván Ruíz Rube,Juan Manuel Dodero y Manuel Palomo Duarte. El proyecto se basa en los mundos virtuales 3D para enseñar el nivel de Alemán A1, mediante una actividad colaborativa de manera similar a la de *Catch me if you can*. Hay que destacar que el equipo de investigación detrás de estos proyectos es prácticamente el mismo.

El juego propone una actividad en las que dos jugadores tendrán que colaborar para resolver un reto en la lengua extranjera. El reto que propone, es el de ordenar debidamente una habitación entre dos jugadores mediante el uso de una conversación por chat. Se valora, el uso del idioma

Figura 2.3: Captura del videojuego OpenSim aplicado al Alemán

por parte del alumno en la actividad, ya que se hace uso de los logs (registros de la conversación) para posteriormente puntuar la actividad. En ese sentido tiene bastante en común con VocabTrainer A1, aunque éste apuesta por los mundos virtuales en vez de hacer una aplicación móvil.

En los mundos virtuales, una gran cantidad de recursos de la máquina en la que funciona el programa se destinan a la ejecución del simulador, y además de esa potencia de procesamiento, también es necesario un servidor que esté soportando la partida en la que se realiza la actividad. Es por esto por lo que aunque se han utilizado mucho durante años y tienen sus grandes ventajas, requieren un gran mantenimiento y un mayor número de recursos comparado al uso de una aplicación móvil que se puede ejecutar en cualquier dispositivo compatible.

El proyecto de aprendizaje ha tenido publicaciones como la de la conferencia de EuroCall 2012.

2.2. Android: ¿Qué es android?

Como apertura, podríamos decir que Android es un sistema operativo para dispositivos móviles. Pero realmente nos estaríamos quedando cortos con esa definición. Android es una modificación del sistema operativo Linux para dispositivos móviles que nos permite tener un sistema multitarea. Esto supone una gran ventaja para los desarrolladores ya que nos proporciona una base sólida para poder desarrollar programas que en sistemas más cerrados no sería posible con la misma facilidad.

Figura 2.4: Logo de Android

Android se presentó oficialmente en 2007 pero no se empezó a usar en terminales móviles por primera vez en 2008. Hasta el año 2011 no fue cuando el sistema consigue despegar y empezar a coger cuota de mercado como alternativa a los móviles de la competencia. Fue entonces cuando se lanzó la versión que durante años (y a día de hoy se sigue usando, sin ir más lejos éste proyecto ha sido probado en un móvil con tal versión) ha predominado en los dispositivos Android, hablamos de GingerBread.

Gingerbread, o Android 2.3, es utilizado como estándar en móviles asequibles debido a sus bajos requisitos y a la compatibilidad que ofrece el sistema. Hoy día los teléfonos con este sistema siguen siendo funcionales aunque empiecen a surgir ciertas aplicaciones incompatibles que requieren un software más actualizado.

Desde aquella versión, Android ha ido comiendo mercado hasta estar presente en alrededor del 90 por ciento de nuestros terminales españoles. Entre la gente joven, y por norma general con poco presupuesto, se sitúa como el sistema más actualizado debido a su bajo precio y al auge de las aplicaciones de mensajería instantánea en pos de los anticuados SMS.

Actualmente, la empresa que está a cargo de desarrollar actualizaciones y de sacar al público novedades respecto a Android, es la gigante de la informática Google. Google, no solo ha ido desarrollando mejoras y lanzamientos de andorid si no que también junto a otras compañías como LG o ASUS han sacado móviles y tablets al mercado que bien pueden competir en rendimiento a otros con un precio que algunas veces roza la mitad.

Debido a estos motivos hoy en día Android es un sistema en auge y que cada vez se utiliza en

Figura 2.5: 'soy un código qr'

más dispositivos alejándose poco a poco de la idea de terminal móvil, ya que ya podemos hoy día comprar relojes inteligentes con tal sistema, o como televisores inteligentes que nos permiten una gama de aplicaciones compatibles. Podemos decir que cada vez tenemos más presente en la sociedad el uso de dispositivos inteligentes (o con el sufijo 'smart' en inglés) para ciertas tareas y no depender tanto de tener un ordenador a mano que debido a sus limitaciones no vamos a llevarlo siempre encima.

Es por eso por lo que hemos elegido esta opción para desarrollar nuestro proyecto, ya que el público al que iba dirigida es un púbico joven y en constante contacto con las nuevas tecnologías, y por que no decirlo, al día con el fenómeno de estar enganchados al móvil con aplicaciones como 'WhatsApp'.

2.3. Zxing, QR y Barcode Scanner

Desde hace un tiempo, se puede observar en nuestro día a día, la existencia de un tipo de código diferente al famoso código de barras. Ese código es el código QR, y viene a decir en inglés respuesta rápida (Quick response). Se compone de una matriz bidimensional para guardar los datos, y a día de hoy todos nuestros móviles (por ejemplo un móvil Android) podrá leerlo.

Fue desarrollado por Denso Wave y proviene de japón, de hecho en japón este código es el más utilizado para almacenar información rápida y aunque fue desarrollado en 1994 no se ha empezado utilizar en nuestro país hasta hace bien poco. Numerosas empresas a día de hoy incluyen códigos qr en sus latas de refresco para dirigir directamente a su web, o para promocionar algún evento.

¿Puedo crear un qr code yo para mi propio uso? Por supuesto, existen páginas webs y numerosas aplicaciones informáticas que te permiten proporcionarles una información y te la convierten a código qr, para que posteriormente podamos hacer el uso apropiado de esa información y colocarla dónde más nos convenga.

Figura 2.6: Logo de la librería Zxing

Es gracias a su facilidad de uso en terminales móviles por lo que elegimos este formato para poder relacionar los objetos propios de la aplicación (por ejemplo, al buscar 'la playa') con un objeto real con éste código impreso. Quiere decir, que podemos dotar de información rápida a cualquier objeto que deseemos podemos 'informatizarlo'.

En la figura (Fig. 2.5) podemos observar un código que almacena una información. Esa información en concreto es la de 'soy un código qr'. Para nosotros ha proporcionado una información en forma de texto que de nosotros dependerá la importancia que tenga, pero bien podría ser una información cifrada para un programa, como por ejemplo una pista de nuestro juego.

Para poder garantizar que el alumno encontraba la pista adecuada en el juego del chat, tuvimos que diseñar códigos para decirle a nuestro programa que pista estaba intentando analizar ademas de mostrar el vídeo correspondiente. De esta manera pudimos informatizar las pistas y separar la información relevante para el alumno (el vídeo) de la información codificada que necesitaba el programa para funcionar.

Para decodificar esa información necesitaremos un lector de códigos que nos codifique por nosotros la información contenida en el código qr. Como el proyecto no trataba de análisis de imágenes y de reconocimiento de patrones, y añadiendo que el código qr es un estándar, vimos como opción idónea utilizar una librería que decodifique el código por nosotros.

Zxing es la librería encargada de hacer eso por nosotros. Con licencia Apache, se convierte en la opción Software libre con mejores resultados que he podido probar. La librería al enlazarla con nuestro proyecto Android nos permitirá lanzar un 'intento' o 'intención' (intent en inglés) de querer analizar el código que tenemos delante.

Claro que para ello, necesitaremos de una aplicación que capture una imágen del código y la guarde en memoria principal. Zxing detectará si tenemos instalada la aplicación y si no la tenemos instalada nos preguntará si queremos instalar la aplicación BarcodeScanner. Si respondemos afirmativamente se nos llevará automáticamente a la página de google play de la

aplicación en concreto para que la podamos instalar con un solo click.

Ésta aplicación también lleva la misma licencia que la librería Zxing y está desarrollada por el mismo equipo (Zxing Team). La aplicación tiene un uso muy sencillo y cuando es invocada por un 'intento' simplemente escaneará el código y proporcionará la información al programa que la haya requerido.

Con la librería y la aplicación, trabajando en conjunto, nuestro proyecto es capaz de tomar la información necesaria desde el exterior y a decisión del alumno, dotándolo así de una gran movilidad y facilidad de uso ya que no requiere ningún conocimiento previo para el usuario.

2.4. Chat, Openfire y Smack

Desde el principio hablábamos de hacer un juego en el que la información no se proporcionara al profesor y el profesor tuviera que dar una respuesta. Queríamos un juego en el que los alumnos tuvieran que cooperar y que la información fluyera entre los estudiantes. De esta manera garantizábamos un mayor aprendizaje en cuanto al desarrollo en grupo de un sencillo misterio a resolver en el idioma que están estudiando.

Nos resultaba más interesante comprobar el nivel de interación en el idioma extranjero entre los alumnos para medir su nivel de desarrollo del idioma escrito en una conversación real. Es ahí dónde entra el chat.

El chat tenía que proporcionarnos la herramienta de comunicación entre los alumnos ya que tenía que ser directa y en tiempo real. Necesitábamos garantizar el envío y la recepción de información por parte de los alumnos dentro del propio juego. Es por ello que el mini juego cooperativo que incluye el proyecto es en su mayor parte un chat para tres personas.

En los chats cada persona adquiere un nombre o 'nick' y en este caso hemos restringido la posibilidad de elegirlo (así como de elegir con quién hablamos) para garantizar el anonimato de la información al menos en el desarrollo de la misma actividad.

A la hora de desarrollar un chat incrustado en nuestra aplicación y por tanto en nuestro sistema Android, no tuvimos que barajar muchas posibilidades ya que de software libre y de implementación 'sencilla' (por sencilla quiero decir que no necesitase de por ejemplo, videoconferencia) pocas opciones había: elegimos Openfire.

Openfire es un conjunto de herramienta-servidor que utiliza el protocolo XMPP para establecer la comunicación de mensajes. Openfire está desarrollado por **Ignite Realtime** y está publicado bajo la licencia Apache de software libre, lo que nos permite usarla en nuestra aplicación.

Por una parte tenemos el servidor, que nos permite tener una base de datos de usuarios registrados que van a hacer uso del juego. El servidor además almacena la información necesaria para

Figura 2.7: Logo del servidor Openfire

el funcionamiento de las salas, necesarias para poder realizar el mini juego en grupos de tres jugadores. El servidor es de fácil administración y también proporciona un conjunto de herramientas y plugins para poder dotarlo de mayor usabilidad, como por ejemplo poder guardar una base de datos de los mensajes de los usuarios.

Cuando un usuario accede con la aplicación mediante su nombre de usuario y contraseña, la aplicación conectará con el servidor para poder validar la sesión, y posteriormente brindarlo de las numerosas opciones que tiene el protocolo XMPP de mensajería.

Pero falta un detalle importante, nuestra aplicación, ¿cómo se conectará al servidor para poder establecer el protocolo de comunicación? Necesitamos una librería que nos facilite el envío de información. En la propia página de Ignite Realtime tenemos a nuestra disposición la librería **Smack** que sirve para dotar a nuestro proyecto Java (hablaremos de él más adelante) con las posibilidades de una aplicación chat.

Si estuviéramos hablando de una aplicación de escritorio para nuestros ordenadores, nos valdría sólo con eso. Pero tenemos que tener en cuenta de que aunque esté programado en el mismo lenguaje que utiliza nuestro sistema Android, no será suficiente ya que existen incompatibilidades con respecto a este sistema. Por ello se precisa de utilizar un 'port' (se dice cuando un programa tiene las mismas características en otro sistema pero ha sido desarrollado posteriormente sin alterar sus prestaciones) de la librería a Android, llamada **Asmack** que viene evidentemente del significado de Android smack.

Este port de la librería junto a su documentación me han permitido desarrollar una aplicación que tuvieras las funciones más comunes de un chat: validar sesión, enviar mensajes, recibir mensajes, mostrar mi estado de conexión, cerrar sesión... entre otras. Como de costumbre con todo lo que hemos utilizado para el proyecto está distribuido bajo la licencia de software libre Apache, por lo que no hay problema en utilizarlo para nuestros menesteres.

2.5. Java Runtime Environment

De las tecnologías presentadas, quizá no la más famosa, pero si la más evidente para el proyecto, es **Java**. Es un lenguaje de programación cuya particularidad es que no es un lenguaje

Figura 2.8: Logo del lenguaje de programación Java

que compile directamente a código máquina. Para las personas que no estén en el mundo de la informática podríamos decir que gracias a este lenguaje de programación no dependemos de estar haciendo un programa para móviles y otro distinto para ordenador. Gracias a este lenguaje, podemos desarrollar una vez y probarlo en diferentes dispositivos.

Java, tiene extendida su máquina virtual (capa de software que es capaz de leer nuestro programa y hacerlo funcionar en la plataforma en la que esté instalada) en numerosos dispositivos como anuncian sus desarrolladores. Si desarrolláramos en otro lenguaje de programación, como por ejemplo el potente C++, tendríamos el inconveniente de tener que adaptar y compilar una vez en Windows, otra en Mac OS y otra en Linux (si así lo quisiéramos). Con Java eso no es necesario ya que será la máquina virtual la encargada de ofrecer la compatibilidad con el sistema en el que se ejecuta nuestro programa.

Dicho esto, ahora se puede justificar el uso de Java en el sistema Android. Java al tener una capa intermedia para la ejecución de sus programas (recordemos, la máquina virtual) hace el proceso de ejecución más lento, pero garantiza su funcionalidad. ¿Hubiese sido posible entonces desarrollar tantísimas aplicaciones para Android en otro lenguaje, y adaptarlo a cada dispositivo con su estructura? La respuesta es que sí, pero tendría un coste muy elevado. La idea de Android era unificar el conjunto de software que funcionara en el sistema.

Esto nos da una grandísima portabilidad, da igual el dispositivo que sea, tablet, móvil, del año que sea, que si desarrollamos una aplicación en Java que no requiera de algo en especial, podremos ejecutarla en cualquier dispositivo. Es por eso que nuestra aplicación está desarrollada en Java bajo Android y por lo que se le debe mencionar en el proyecto en contraposición con otras tecnologías.

Capítulo 3

Planificación

3.1. Planificación

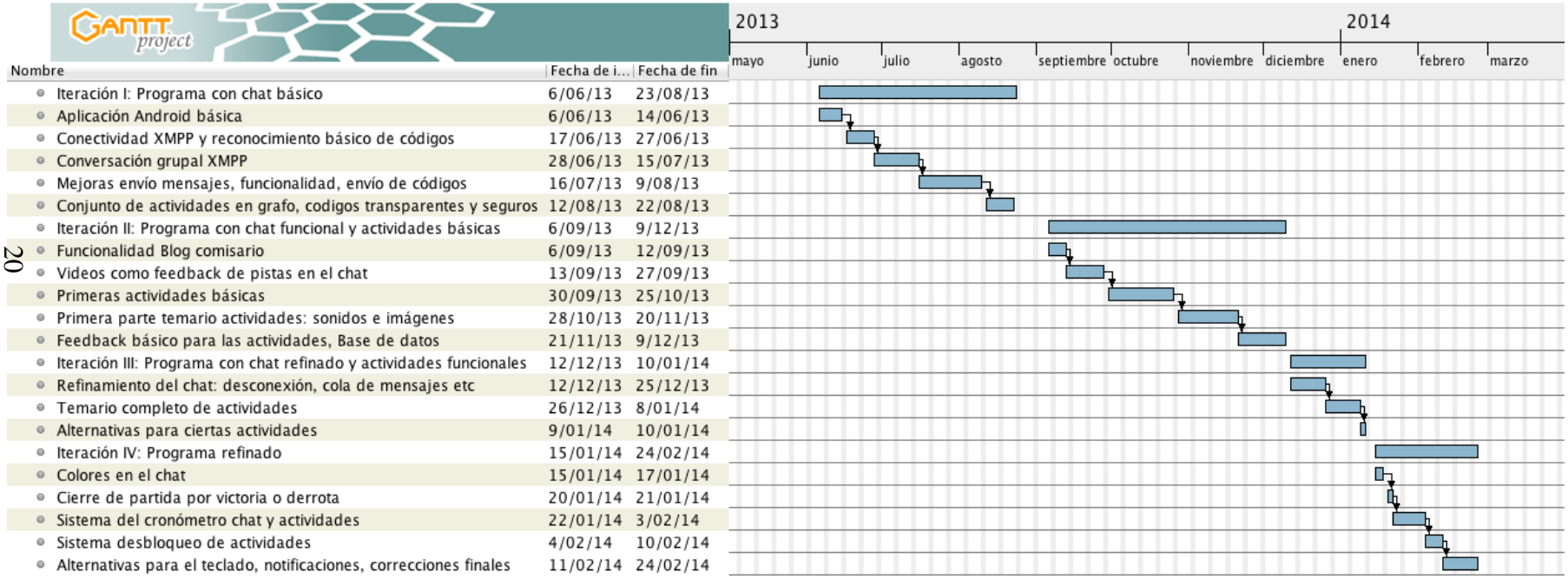
El proyecto empezó a gestarse a finales de Mayo de 2013. No fue hasta mediados de Junio cuando teníamos planteados los requisitos de la aplicación que queríamos desarrollar. Como estimación de tiempo queríamos tener la aplicación lista para principios de febrero, para que el PFC se entregase en un tiempo antes del verano del año 2014, y para que se pudiese empezar a utilizar a principios de ese mismo año.

Nos planificamos que realizar un proyecto de esta envergadura, se tardarían unos 5 o 6 meses en desarrollarse, ya que queríamos añadirle un temario mediante el cual sustentar las actividades y que requería de un tiempo amplio también. Planteamos cuatro iteraciones en las que nos llevaríamos de tiempo para cada una de ellas de uno a dos meses (uno para las más sencillas y dos para las más complejas).

Finalmente, la planificación quedó como en la figura 3.1, dónde se puede apreciar el gran esfuerzo de tiempo que supuso las dos primeras iteraciones. En el caso de la segunda llegó a ocupar tres meses de nuestro tiempo. Esto se debe en gran parte a la investigación de la tecnología de Android y a la del protocolo XMPP, dónde cada paso que dábamos era un paso nuevo. Las dos últimas iteraciones fueron más cortas ya que teníamos una buena base de la aplicación y podíamos añadir funcionalidades de manera más directa y sencilla. Aparte una vez creadas las lógicas de las actividades y del chat lo siguiente era refinarlas y dotarlas de temario.

Un punto a destacar fue que durante el comienzo del desarrollo hubo que cambiar de concepto en varias ocasiones y por tanto se alargó bastante más el desarrollo de las primeras iteraciones del sistema. Aunque planificamos que las iteraciones no duraran mucho más de dos meses en la primera hubo que hacer un gran esfuerzo de investigación para ver que queríamos hacer, y en la segunda se dedicó gran parte de desarrollo al contenido de la aplicación y a la depuración de todo el sistema, lo que nos llevó más tiempo del que creíamos. La parte positiva es que la tercera y cuarta iteración duró menos de lo esperado.

A continuación pasamos a explicar cada iteración y sus características.



Nombre	Fecha de i...	Fecha de fin
Iteración I: Programa con chat básico	6/06/13	23/08/13
Aplicación Android básica	6/06/13	14/06/13
Conectividad XMPP y reconocimiento básico de códigos	17/06/13	27/06/13
Conversación grupal XMPP	28/06/13	15/07/13
Mejoras envío mensajes, funcionalidad, envío de códigos	16/07/13	9/08/13
Conjunto de actividades en grafo, codigos transparentes y seguros	12/08/13	22/08/13
Iteración II: Programa con chat funcional y actividades básicas	6/09/13	9/12/13
Funcionalidad Blog comisario	6/09/13	12/09/13
Videos como feedback de pistas en el chat	13/09/13	27/09/13
Primeras actividades básicas	30/09/13	25/10/13
Primera parte temario actividades: sonidos e imágenes	28/10/13	20/11/13
Feedback básico para las actividades, Base de datos	21/11/13	9/12/13
Iteración III: Programa con chat refinado y actividades funcionales	12/12/13	10/01/14
Refinamiento del chat: desconexión, cola de mensajes etc	12/12/13	25/12/13
Temario completo de actividades	26/12/13	8/01/14
Alternativas para ciertas actividades	9/01/14	10/01/14
Iteración IV: Programa refinado	15/01/14	24/02/14
Colores en el chat	15/01/14	17/01/14
Cierre de partida por victoria o derrota	20/01/14	21/01/14
Sistema del cronómetro chat y actividades	22/01/14	3/02/14
Sistema desbloqueo de actividades	4/02/14	10/02/14
Alternativas para el teclado, notificaciones, correcciones finales	11/02/14	24/02/14

Figura 3.1: Diagrama de Gantt del proyecto

3.2. Iteración I: Programa con chat básico

La primera iteración del programa tenía como requisito que se pudiera tener una prueba de concepto del chat. Para nosotros fue importante antes de desarrollar cada aspecto de la actividad colaborativa poder ver como podían comunicarse los alumnos a través del móvil. Por ello la aplicación debía tener un chat que permitiese las características mínimas de una aplicación de mensajería instantánea.

Esta iteración, tenía la posibilidad de poder conectarse al servidor con usuario y contraseña, permitía el envío y recepción de mensajes en la sala creada para ello y la posibilidad de desconectarse. Esta versión del programa tenía bastantes fallos y el chat no era muy estable a la hora de usar ciertas funciones como el giro de pantalla del dispositivo.

Uno de sus aspectos más importantes era que la manera en la que se enviaban los códigos escaneados por mensajería instantánea. Se enviaban de manera completamente transparente para el alumno y de manera codificada para que el alumno no pudiese alterar su funcionamiento.

3.3. Iteración II: Programa con chat funcional y actividades básicas

Esta segunda iteración fue la más larga de desarrollar y es debido básicamente a sus requisitos. Necesitabamos que el chat tuviera ya todas sus funciones funcionando correctamente para que se pudiese jugar con los dos roles del juego *Catch me if you can*, y también era necesario que los alumnos de prueba tuvieran acceso a las actividades para que pudiera repasar parte del temario (ya que aún no estaba completo y faltaban funciones), aunque todavía no era requisito necesario para poder jugar en el chat.

La versión contenía como características:

- Chat con blog de notas del comisario.
- Reproducción de vídeos al escanear un código mientras se juega en el chat.
- Actividades separadas por temario.
- Sonido e imágenes para las actividades.
- Añadido feedback en las actividades.

3.4. Iteración III: Programa con chat refinado y actividades funcionales

En esta iteración ya teníamos la experiencia de haber realizado una prueba con alumnos (al final de la iteración dos) y pudimos corregir ciertos aspectos que eran necesarios para el correcto funcionamiento del chat. Aparte añadimos como requisito tener que terminar todas las actividades para poder acceder al chat.

Esta versión ya tenía un conjunto de actividades con todo su temario y permitía probar la experiencia que queríamos conseguir con la App.

Sus características son:

- Mejora de la estabilidad del chat.
- Sistema de detección de pérdida de conexión en el chat.
- Cola de mensajes offline.
- Reconexión automática si se pierde la conexión en el chat.
- Añadida base de datos en la que la aplicación guardaba el progreso de las partidas para ir desbloqueando actividades (era necesario terminar todas las actividades para acceder al chat).
- Añadida la posibilidad de borrar tus partidas para dar la posibilidad de empezar de nuevo.
- Corregidos diversos aspectos de las actividades.
- Temarios completos para las actividades.
- Mejorado feedback de las actividades dando la posibilidad de mostrar varias alternativas.

3.5. Iteración IV: Programa refinado

Esta iteración ya contiene la versión final del programa. En esta versión dedicamos nuestro esfuerzo en mejorar todos los aspectos que ya estaban desarrollados en las anteriores iteraciones. Mejoramos los retos de las actividades con la adicción del tiempo como hándicap, nos centramos en mejorar la experiencia de juego tanto en el modo en solitario como en el modo colaborativo.

Sus grandes mejoras frente a la anterior iteración son:

- Añadido tiempo en todas las actividades.

– Colores para distinguir a cada participante del chat y mensajes de aviso en verde/rojo.
– Añadido cierre de la partida *Catch me if you can* al terminar el tiempo o al ganar la partida.
– Corregidos bugs que ocurrían a la hora de guardar y cargar la partida del perfil.
– Añadidos diversos avisos que saltan al terminar una partida.
– Refinado el feedback añadiendo el tiempo utilizado para terminar una partida.
– Añadida alternativa para usuarios sin los caracteres ü y ä.

Parte II

Desarrollo

Capítulo 4

Análisis

4.1. Análisis del sistema

En esta sección procederemos a hacer un análisis del sistema, mostrando los diferentes casos de uso que tiene la aplicación y mostrando detalladamente el diagrama de clases del propio sistema.

4.1.1. Diagramas de análisis

Además en las (Fig. 4.1) y (Fig. 4.2) se pueden observar los diferentes diagramas pertenecientes al flujo del juego y al método de asignación de roles a la hora de jugar colaborativamente.

4.1.2. Modelo de casos de uso

En la (Fig. 4.3) tenemos la imagen que describe el Modelo de casos de uso del sistema. Pasamos a describir detalladamente cada caso.

4.1.3. Caso de uso: Introducir nombre

Escenario principal

- El usuario pulsa sobre el botón Perfil en el menú del sistema.
- El sistema carga la vista de Perfil.
- El usuario escribe sobre el espacio designado para escribir el nombre y pulsa enviar.
- El sistema asigna el nombre de usuario escrito al perfil y muestra la vista del menú del sistema.

Escenario alternativo I

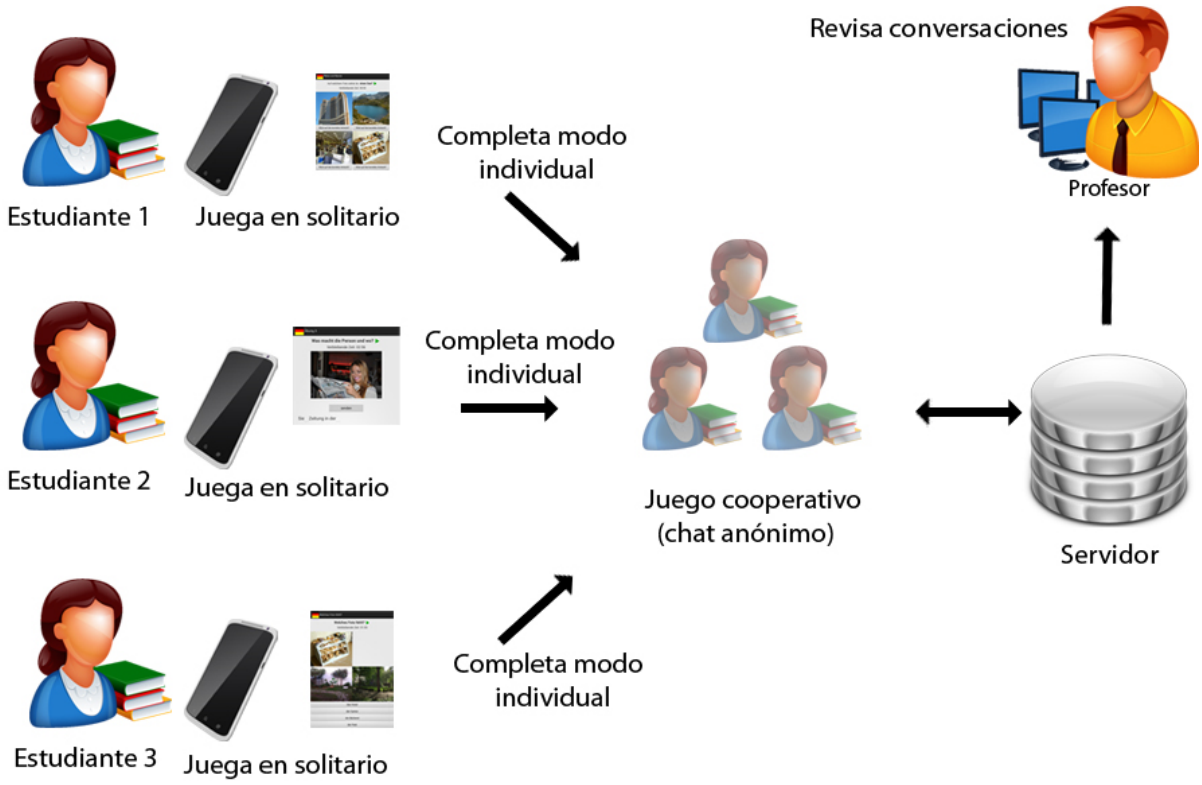


Figura 4.1: Diagrama que muestra la estructura del juego

Figura 4.2: Diagrama que muestra la asignación de roles de los jugadores

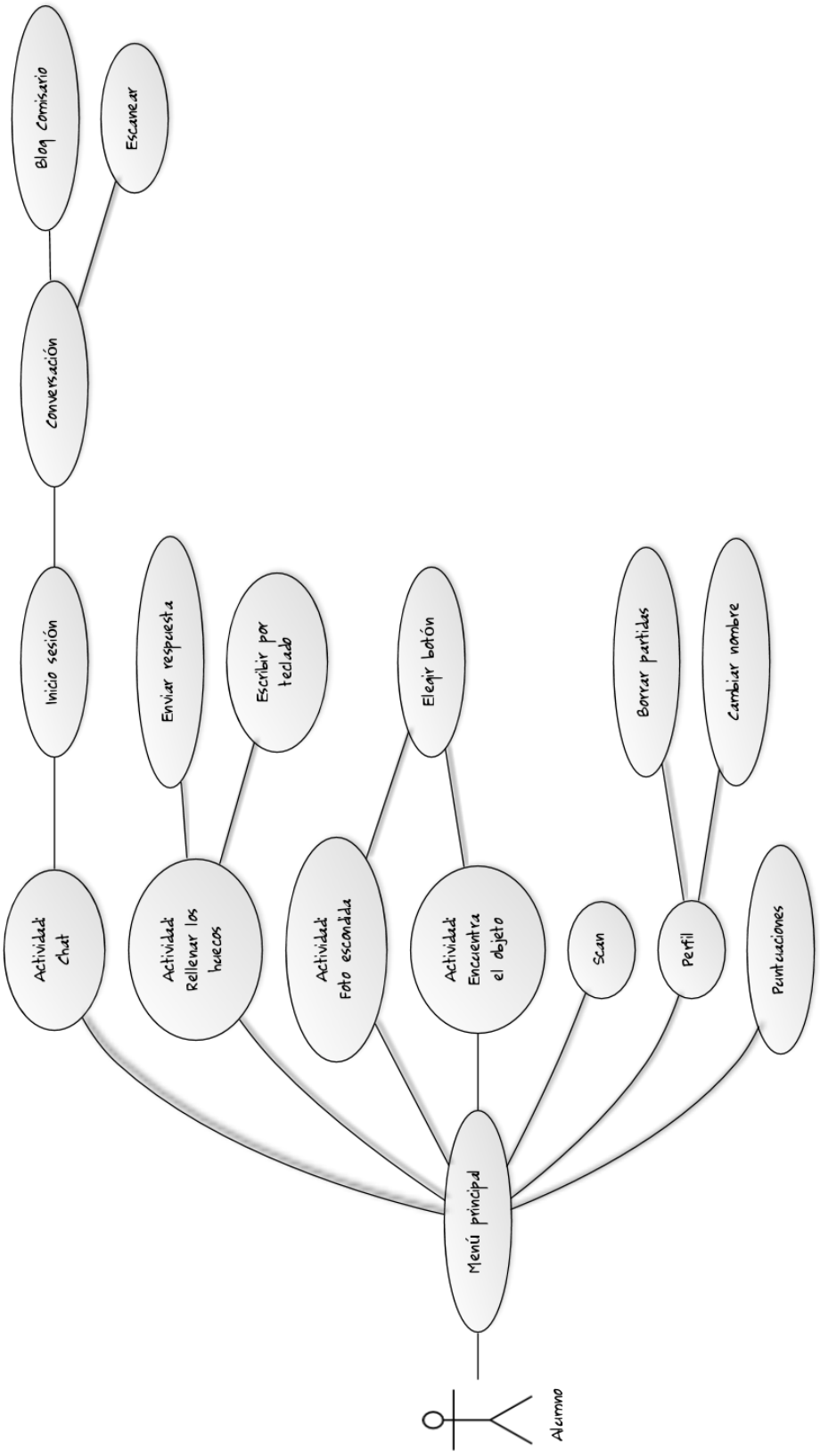


Figura 4.3: Modelo de casos de uso

- El usuario pulsa sobre el botón Perfil en el menú del sistema.
- El sistema carga la vista de Perfil.
- El usuario escribe un nombre de usuario y pulsa el botón atrás de Android.
- El sistema cierra la vista de Perfil sin cambiar el nombre de usuario y muestra la vista del Menú principal.

4.1.4. Caso de uso: Borrar partidas

Escenario principal

- El usuario pulsa sobre el botón Perfil en el menú del sistema.
- El sistema carga la vista de Perfil.
- El usuario pulsa el botón Resetear perfil y puntuaciones.
- El sistema muestra un diálogo de confirmación.
- El usuario acepta el diálogo.
- El sistema borra tanto el nombre del jugador como sus partidas, muestra un mensaje de 'Perfil borrado' y carga la vista del Menú principal.

Escenario alternativo I

- El usuario pulsa sobre el botón Perfil en el menú del sistema.
- El sistema carga la vista de Perfil.
- El usuario pulsa el botón Resetear perfil y puntuaciones.
- El sistema muestra un diálogo de confirmación.
- El usuario rechaza el diálogo y pulsa el botón atrás de Android.
- El sistema carga la vista del Menú principal y no realiza ningún cambio en el perfil.

4.1.5. Caso de uso: Ver puntuaciones

Escenario principal

- El usuario pulsa sobre el botón Puntuaciones en el menú del sistema.
- El sistema carga la vista de las Puntuaciones y muestra las puntuaciones obtenidas por cada Actividad.
- El usuario pulsa el botón atrás de Android.
- El sistema carga la vista del Menú principal.

4.1.6. Caso de uso: Reproducir sonido de ayuda

Escenario principal

- Se muestra la actividad tipo 1, 2 o 3.
- El usuario pulsa el botón reproducir sonido de ayuda.
- El sistema reproduce el sonido correspondiente a la ayuda de la actividad.

4.1.7. Caso de uso: Elegir foto

Escenario principal

- Se muestra la actividad tipo 1 o 2.
- El usuario pulsa el botón de la foto indicada.
- El sistema muestra durante unos segundos el botón en color verde y reproduce el sonido asociado a la foto.
- El sistema carga el siguiente conjunto de fotos.

Escenario alternativo I

- Se muestra la actividad tipo 1 o 2.
- El usuario pulsa el botón de la foto errónea.
- El sistema deja el botón en color rojo y desactivado y reproduce el sonido asociado a la foto.

Escenario alternativo II

- Se muestra la actividad tipo 1 o 2.
- El usuario pulsa el botón atrás de Android
- El sistema carga la vista del Menú principal.

4.1.8. Caso de uso: Rellenar el hueco

Escenario principal

- Se muestra la actividad tipo 3.
- El usuario escribe correctamente la respuesta y pulsa el botón enviar.
- El sistema reproduce un sonido indicando que está correcto y carga la siguiente foto.

Escenario alternativo I

- Se muestra la actividad tipo 3.
- El usuario escribe erróneamente la respuesta y pulsa enviar.
- El sistema avisa mediante texto de que la respuesta es errónea.

Escenario alternativo II

- Se muestra la actividad tipo 3.
- El usuario escribe erróneamente la respuesta después de dos intentos y pulsa enviar.
- El sistema muestra la respuesta correcta para que el jugador pueda seguir avanzando.

4.1.9. Caso de uso: Iniciar sesión

Escenario principal

- El usuario pulsa sobre el botón Chat en el menú principal.
- El sistema establece conexión y carga la vista del inicio de sesión.
- El usuario escribe su nombre de usuario y contraseña y pulsa el botón Login.
- El sistema consigue iniciar sesión y muestra la vista de empezar la partida del Chat.

Escenario alternativo I

- El usuario pulsa sobre el botón Chat en el menú principal.
- El sistema establece conexión y carga la vista del inicio de sesión.
- El usuario escribe su nombre de usuario y contraseña y pulsa el botón Login.
- El sistema no consigue iniciar sesión por que los datos no son correctos y muestra un aviso por pantalla.

Escenario alternativo II

- El usuario pulsa sobre el botón Chat en el menú principal.
- El sistema establece conexión y carga la vista del inicio de sesión.
- El usuario pulsa el botón atrás de Android.
- El sistema desconecta con el servidor y carga la vista del Menú principal.

4.1.10. Caso de uso: Empezar partida Chat

Escenario principal

- El sistema muestra la vista de empezar la partida.
- El usuario pulsa sobre el botón Comenzar partida.
- El sistema crea una sala y asigna el rol de comisario al jugador. El sistema carga la vista de la conversación.

Escenario alternativo I

- El sistema muestra la vista de empezar la partida.
- El usuario pulsa sobre el botón Comenzar partida.
- El sistema encuentra a otros jugadores jugando en una sala y se une a ella, asignando al usuario el rol de policía.
- El sistema carga la vista de la conversación.

Escenario alternativo II

- El sistema muestra la vista de empezar la partida.
- El usuario pulsa sobre el botón Salir o el botón atrás de Android.
- El sistema cierra sesión y desconecta del chat.
- El sistema muestra la vista del Menú principal y muestra un mensaje de 'Desconectado'.

4.1.11. Caso de uso: Escribir en el blog del comisario

Escenario principal

- El sistema muestra la vista de conversación del chat.
- El usuario pulsa sobre el botón Blog.
- El sistema carga la vista del blog.
- El usuario escribe en cualquiera de los campos para la descripción del 'asesino'.
- El usuario pulsa el botón atrás de Android o pulsa en guardar blog.
- El sistema guarda las descripciones descritas por el usuario y muestra la vista de la conversación.

Escenario alternativo I

- El sistema muestra la vista de conversación del chat.
- El usuario pulsa sobre el botón Blog.
- El sistema carga la vista del blog.
- El usuario pulsa el botón atrás de Android o pulsa en guardar blog.
- El sistema carga la vista de la conversación.

4.1.12. Caso de uso: Escanear código

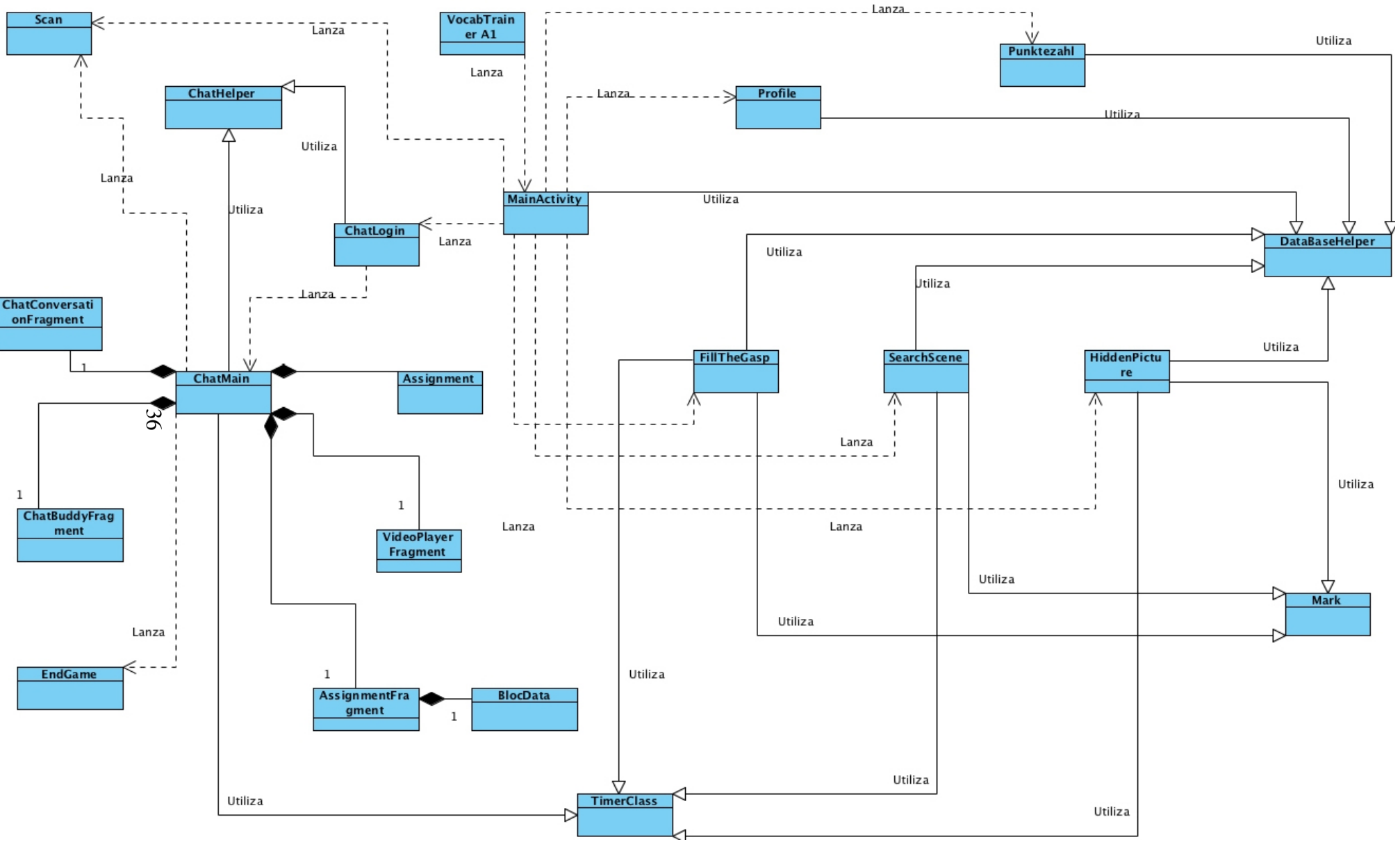
Escenario principal

- El sistema muestra la vista de conversación del chat.
- El usuario pulsa sobre el botón Scan.
- El sistema carga la cámara para realizar un escaneo.
- El usuario acerca la cámara de su móvil a un código y logra enfocarlo.
- El sistema escanea el código y envía el código cifrado por la conversación.
- El sistema muestra el vídeo del código que acabamos de escanear si el usuario tiene el rol de policía, en caso contrario muestra la conversación con el resultado del escaneo.

Escenario alternativo I

- El sistema muestra la vista de conversación del chat.
- El usuario pulsa sobre el botón Scan.
- El sistema carga la cámara para realizar un escaneo.
- El usuario pulsa el botón atrás de Android.
- El sistema carga la vista de la conversación.

4.1.13. Diagrama de clases del sistema


Scan
VocabTrainer A1
Punktezahl
ChatHelper
Profile
MainActivity
ChatLogin
DataBaseHelper
ChatConversationFragment
ChatMain
Assignment
FillTheGasp
SearchScene
HiddenPicture
ChatBuddyFragment
VideoPlayerFragment
Mark
EndGame
AssignmentFragment
BlocData
TimerClass
Lanza
Utiliza
1
36

Capítulo 5

Diseño

5.1. Software utilizado

En esta sección paso a comentar todo el software utilizado para el desarrollo la aplicación, así como sus materiales de sonido, vídeo e imágenes.

He tenido que utilizar un gran abanico de herramientas ya que algunas me proporcionaban parte de lo que necesitaba pero necesitando de una segunda aplicación para poder convertir un vídeo al formato propio de Android.

5.1.1. Eclipse

Eclipse es uno de los mayores IDE (entornos de desarrollo) que existen en el mercado de software. Gracias a los entornos de desarrollo somos capaces de mediante un lenguaje (o lenguajes) de programación, poder desarrollar aplicaciones y tener a mano funcionalidades extras como control de versiones, editor de ficheros, manejo de errores, depuradores... etc.

Fue desarrollado inicialmente por **IBM** para más tarde pasar a ser de la Eclipse Foundation,

Figura 5.1: Logo de Eclipse

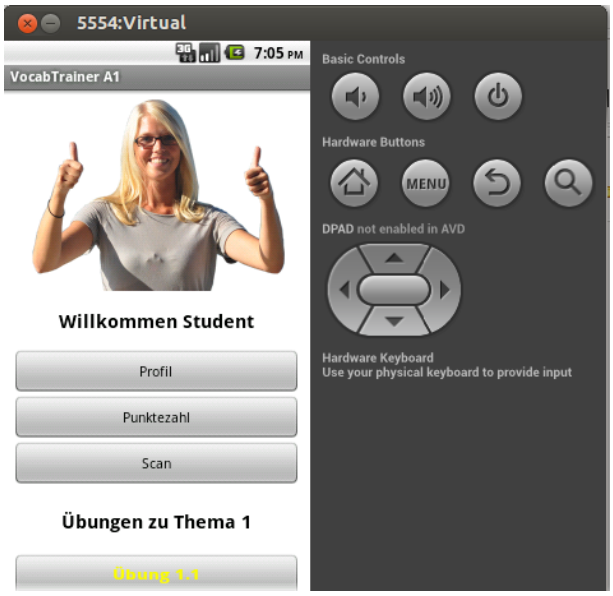


Figura 5.2: VocabTrainer A1 corriendo en el emulador

quién acabó liberando el código finalmente mediante su propia licencia (Eclipse Public License). Éste IDE nos permite poder desarrollar aplicaciones en Java (un casi 93 por ciento de las veces se usa Eclipse para éste lenguaje) aunque también se puede utilizar para desarrollar en otros lenguajes como C++ o php. Tiene una gran compatibilidad ya que funciona perfectamente en Windows Linux y Mac OS pudiendo desarrollar para las tres plataformas por igual.

La importancia de este programa para mi proyecto ha sido **vital** ya que gracias al gran conjunto de herramientas y plugins (añadidos) de las que se dispone al instalarlo, me han hecho posible poder desarrollar y probar mi proyecto para Android en un mismo sistema. Para el desarrollo del mismo he utilizado el lenguaje de programación Java junto al plugin que describiremos en el siguiente punto, el Android Development Tools plugin (ADT).

5.1.2. Android SDK

Aunque el desarrollo de la aplicación se ejecute en código Java, es necesario un conjunto de herramientas para poder hacerlo funcionar en Android. Necesitaríamos una biblioteca de clases programadas previamente para el sistema para tener una interfaz propia del sistema, un depurador que nos otorgue el control de lo que está ocurriendo desde una terminal y por si fuese necesario (no todo el mundo tiene un dispositivo Android) un simulador en el que probar el comportamiento simulado de nuestra aplicación.

Android SDK contiene lo descrito en el párrafo anterior, y como IDE soportado **oficialmente** tiene el descrito con anterioridad: Eclipse. Para poder integrar el Android SDK en Eclipse, utilizaremos el plugin ADT. Una vez integrado el plugin en nuestro IDE, podremos crear un

Figura 5.3: Logo de Gimp

nuevo proyecto Android con mucha facilidad. El entorno nos creará una jerarquía de ficheros necesarios para empezar a programar nuestra aplicación.

Desarrollaremos nuestras clases Java en el apartado de código del proyecto y tendremos a nuestra disposición la librería de Android que nos permitirá incluir imágenes, vídeos, texto o sonidos a nuestra aplicación con mucha facilidad para el desarrollador. A la hora de ejecutar nuestro proyecto, elegiremos entre ejecutarlo en un dispositivo conectado al equipo mediante USB, o podremos simular un dispositivo en el escritorio.

La emulación de Android es a menudo muy costosa para el sistema (en mi caso ralentizaba muchisimo el sistema con un doble núcleo y 2GB de memoria RAM) pero podremos elegir muchas de las características del teléfono como por ejemplo su dimensión de pantalla, su memoria RAM e incluso si queremos emular una tarjeta SD. Esa opción es muy potente ya que podríamos comprobar como queda nuestra aplicación en una tablet de 10 pulgadas y un buen procesador sin tener que adquirir ningún dispositivo.

Además podremos elegir la versión para la que estará enfocada nuestro dispositivo. Por ejemplo en el caso de mi aplicación se eligió como versión mínima Android 2.2 (aunque mi dispositivo y el objetivo era el 2.3) y pude probarlo mediante emulador en un dispositivo virtual para comprobar si existían ciertas incompatibilidades (que las había).

5.1.3. Gimp

Un programa de gran utilidad, conocido por pocos fuera del mundo del desarrollo de software ya que vive a la sombra del gigante de Adobe, Photoshop. Como alternativa de edición de imágenes muestra su lado positivo en que es de Software libre (y gracias a ello está en constante evolución) y a que es gratis. En esta sección repetiremos varias veces la palabra gratis, pero cuando hablamos de que la alternativa más conocida cuesta unos 1.400 euros creo que es un punto a destacar.

El editor nos permite como su directo competidor, una edición de imágenes gestionada por capas en las que poder diferenciar entre texturas y poder aplicarles diversos filtros y modificaciones de manera independiente. Cualquier persona con conocimientos de edición de imágenes y que tenga una base de uso con el programa será capaz de hacer fotomontajes profesionales.

Para mi proyecto necesitaba un editor con el que poder recortar las imágenes, cambiarles la resolución y comprimirlas a una calidad inferior ya que, en la pantalla de un móvil pequeño no seremos capaces de apreciar mucha diferencia entre un fichero de 50 KB o uno de 100 KB, y dado que necesitábamos optimizar al máximo el tamaño de la app Gimp me proporcionó lo que necesitaba.

5.1.4. Pitivi

Dado que tuvimos que comprimir lo máximo posible sin tener pérdidas notables en la reproducción de vídeos, investigué bastantes editores de vídeo para poder cortar, redimensionar y bajar el *bitrate* (tasa de bits por segundo que tiene el vídeo, a más tasa más ocupará el fichero al compilar).

Elegí **Pitivi** ya que me resultó el más sencillo de usar e instalar ya que uno de los dos entornos que he utilizado para desarrollar la aplicación ha sido Ubuntu y fue inmediato instalar la aplicación.

El programa dispone de la típica linea temporal donde editar tus clips de vídeos para luego poder compilarlos cambiando las características generales de los vídeos. Pitivi está distribuido bajo la licencia LGPL lo que le hace una buena alternativa a la edición de vídeo en el ámbito del Software Libre, y además es de uso gratuito.

5.1.5. Handbrake

En este caso **Handbrake** es ligeramente diferente al anterior. También se trata de un editor de vídeo, pero no destaca por tener una línea temporal de vídeo. Este programa tiene su utilidad en el manejo de los detalles con los que vamos a codificar el vídeo.

Figura 5.4: Captura de la aplicación

Estuve investigando bastantes opciones y sólo Handbrake me dio la posibilidad de poder modificar los perfiles del codec h264. Como comentaré más adelante era estrictamente necesario poder dotar de unas ciertas características a los vídeos de la aplicación, ya que aunque ya ocupaban el espacio que queríamos habiendo usado Pitivi primero, necesitaba poder usar la característica *Baseline profile* para el correcto funcionamiento de los vídeos en la aplicación.

Como otro punto positivo destaco que el programa es Software libre distribuido bajo la licencia GPL, y está disponible en las tres principales plataformas (Linux, Mac OS y Windows). Aparte y como destaca la página en inglés Ït's free"(es decir, gratis).

5.1.6. Audacity

Audacity es un programa con muchos años de experiencia detrás (desde el año 2000), en el que se ha consolidado como uno de los programas de edición de sonido favoritos. Distribuido bajo la licencia GPL es una gran opción para la edición tanto profesional como amateur de sonido desde Mac, Windows o Linux.

El programa se presenta bajo la conocida interfaz de edición línea de tiempo, donde podemos importar nuestros clips de audio para poder recortarlos, aplicarle filtros, comprimirlos e incluso modificar sus valores como la amplitud de onda... etc. La utilidad a la que le saqué partido fue a poder recortar fácilmente los clips de sonido, aumentarles su amplitud (tenemos que tener en

Figura 5.5: Logo de Audacity

Figura 5.6: Logo de Git

cuenta de que lo reproduciremos en un móvil en lugares con ruido ambiente) y a poder comprimirlos para sacarles una ganancia de tamaño importante.

Aparte también nos permite grabar clips de audio directamente desde un micrófono por lo que tendremos una potente herramienta para crear materiales para nuestra aplicación.

5.1.7. Git y eGit

Git es el software de control de versiones que he elegido para el desarrollo del proyecto. Fue desarrollado por Linus Torvalds (creador del sistema operativo Linux) y a día de hoy se usa en grandes proyectos como puede ser el propio núcleo de Linux. Como es costumbre en el desarrollo de éste proyecto, Git es software libre distribuido bajo la licencia GPL, lo que lo hace una alternativa libre de usar.

Git sirve para tener copia de seguridad de tu código en un repositorio (lugar dónde se almacenará tu código) y poder acceder a él remotamente, así como poder restaurar una versión concreta en caso de que la actual deba ser remplazada por una anterior.

Para integrarlo con Eclipse, me descargué el plugin EGit que te permite tener el control de las versiones de tu código desde dentro del propio IDE, sin tener que usar otra aplicación o usar comandos de terminal.

5.1.8. Repositorio

Para poder alojar el código queríamos tener un repositorio privado durante el desarrollo del proyecto, que nos sirviera para hacer pruebas y tener el control del código. Para ello utilizamos el repositorio Bitbucket. El proyecto está alojado en la web: `https://bitbucket.org/manuelpfc/deustchuca` Como parte negativa señalo que solo permite 5 participantes en un proyecto privado, pero que es una de las mejores opciones a tener en cuenta ya que no requiere cuotas.

Como parte positiva tiene espacio para descargas, espacio para wiki, y que poco a poco empieza a tener traducción de la interfaz al Español.

* Datos extraídos de la web Wikipedia

5.2. Diseño de los Layout

5.2.1. Contenedores

Los layout son los ficheros xml con los que se diseñan las vistas de las ventanas de nuestra aplicación Android. Cada vista contendrá contenedores del estilo LinearLayout, RelativeLayout o ScrollView entre otros. Cada uno tendrá unas características propias del contenedor, el contenedor Linear nos dará una estructura más lineal, mientras que con el relativo podremos incluir objetos de manera realtiva a otros.

En mi proyecto he escogido el ScrollView para poder adaptarme bien a cualquier pantalla. Con el ScrollView solo podremos contener un solo objeto (en mi caso otro LinearLayout) pero nos permitirá hacer scroll sobre él. En otras palabras, quise que si una vista no cabía en nuestro móvil, pudiesemos usar el dedo para arrastrar lo que hubiese debajo. Un claro ejemplo es la agenda de un teléfono Android. Se muestran un conjunto de contactos, por ejemplo unos 10, pero realmente en nuestro telefono tenemos a 80 contactos. Tendremos pues que arrastrar con el dedo la imagen de nuestro teléfono para ir mostrando secuencialmente los 70 contactos restantes.

Dado que pensé la aplicación para que se pudiera usar en el mayor número de dispositivos posibles, aparte de adaptar a dos resoluciones diferentes, la mejor idea era usar una estructura ScrollView, ya que en las pruebas a veces ciertos botones o imágenes quedaban fuera de la pantalla del dispositivo. Actualmente la aplicación no presenta ningún problema para ejecutarse en una pantalla de 3 pulgadas o en una de 7, ya que aunque no se pueda abarcar la imagen en su totalidad siempre podemos arrastrar la parte no visible.

5.2.2. Resoluciones y tamaños de pantalla

Para mantener la compatibilidad con los dispositivos de todas las gamas, Android proporciona una serie de directorios en los que poder adaptar tu aplicación.

Primero tenemos que diferenciar dos puntos diferentes. Tenemos por un lado la resolución de la pantalla y por otro lado el tamaño de la pantalla. A más resolución de pantalla, tendremos más cantidad de píxeles por pantalla, a más pulgadas más grande será la pantalla. En ese caso Android nos recomienda utilizar recursos (imágenes por ejemplo) dependiendo de la resolución de la pantalla y vistas dependiendo del tamaño de la pantalla.

- Resoluciones: Si nuestra pantalla tiene una resolución de 1280x720 (HD ready) y mostramos a pantalla completa una imagen de 600x400, se verá de baja calidad ya que su densidad de píxeles es muy baja comparada con la de nuestra pantalla. Por tanto debemos de hacer dos ficheros, uno en resolución media y otro en resolución alta para estos casos. El primero irá alojado en la carpeta drawable-mdpi y el segundo en la carpeta drawable-hdpi.

  En nuestro caso, la aplicación necesita de optimizar al máximo su tamaño en MB ya que tenemos muchos materiales de vídeo y sonido que mostrar, por lo tanto, la mejor opción fue **no** diferenciar entre las diferentes resoluciones y poner imágenes de baja resolución, ya que éstas se adaptan bien en los móviles de gama baja y son las que menos ocupan frente a las de alta resolución. ¿Supone esto un problema? a efectos prácticos de funcionalidad ninguno, pero si usamos un dispositivo con mucha resolución notaremos que la imagen no tiene la calidad adecuada para nuestra pantalla (es decir, que se verá pixelada).

- Tamaños de pantalla: Dependiendo del tamaño de la pantalla, deberemos ajustar los elementos mostrados (vistas, textos) para que queden de manera correcta en la imagen de nuestra pantalla. Para ello Android proporciona una jerarquía de carpetas en las que definir la vista (layout) más adecuada para nuestra pantalla. Ya que esto no supone un aumento considerable de memoria, esta vez **sí** hemos diferenciado entre pantallas pequeñas y pantallas con muchas pulgadas.

  Cuando tenemos una pantalla de 7 pulgadas, veremos que algunos textos quedan pequeños y hay muchos 'vacío' en la vista de la aplicación. En cambio esa misma vista cargada en un móvil puede suponer que la frase empiece en un renglón y acabe cortándose cuatro renglones por debajo. En mi caso, tuve que adaptar los tamaños de letra en dos versiones, normal (para dispositivos de 3 a 5 pulgadas) y la larga (para tablets).

  En la vista de las tablets todo está organizado para sacar más provecho de la pantalla que tenemos a nuestra disposición y por tanto no hará falta utilizar tanto el Scroll para poder mostrar todo (aunque esto dependerá de varios factores). Esta vista muestra las imágenes

y vídeos en un tamaño más grande que hace muy cómodo su uso.

En cambio la vista de los móviles está optimizada para no perder funcionalidad. La vista está reorganizada de modo que los textos se puedan leer a un tamaño de letra en el que no haya que forzar la vista, y las imágenes se muestran ligeramente más pequeñas para poder encajar en el tamaño de la pantalla, así como los botones. En este tipo de vista si será necesario en muchos casos hacer uso del Scroll, ya que no nos será posible muchas veces mostrar las cuatro imágenes de una actividad con sus respectivos cuatro botones sin perder la homogeneidad.

5.3. Temarios

La aplicación está dividida en tres temarios con el propósito de poder diferenciar el aprendizaje por dificultades en el idioma. A su vez, los temarios están compuestos por un vocabulario en concreto que se desarrollará en tres partes o actividades. Para poder dar un poco de variedad y poder ajustar dificultad a la aplicación, se desarrollaron tres temarios con su vocabulario propio dónde el primero trata sobre lugares y acciones comunes, el segundo de objetos y el tercero de características de las personas.

Éstos tres temarios suponen la mayor parte de contenido de la aplicación y son los cimientos para poder desarrollar bien una actividad en el idioma extranjero. Para poder acceder al chat es requisito imprescindible haber superado los siguientes tres temarios:

En el **primer temario** tratamos los **lugares** ya que serán los lugares que tendremos que saber reconocer en la actividad grupal. El manejo de los lugares debe ser intuitivo para poder ir de un lugar a otro reconociendo dónde pueda estar un testigo y que hace concretamente. Éste primer temario resulta más sencillo que los siguientes y por ello está colocado el primero, a parte de que como introducción al repaso de un A1 es importante tratar los lugares y las acciones. Acciones que serán tan comunes como escuchar música, charla, o leer.

En el **segundo temario** se tratará los **objetos** que podrían aparecer en la última actividad *catch me if you can* y que suelen estar presentes en la vida cotidiana o en las obras de cine policíaco. Éste temario sirve para diferenciar los objetos que pueda usar el *asesino ficticio* en la última actividad y dado que tiene una mayor dificultad saber diferenciarlos está posicionado como el segundo temario. Aparte tendremos que ser capaces de diferenciar cuando escribir *Eine ein* o *einen* delante de un sustantivo.

En el **tercer temario** (y último) tendremos las **características** más llamativas de las personas como vocabulario. Se tratarán aspectos como el color del pelo, el tamaño de la nariz, si tiene un tatuaje... y supone el paso determinante para poder diferenciar entre los distintas apariencias de los *asesinos ficticios* que aparecen en la última actividad colaborativa. Si el alumno no ha superado estos test no será capaz de garantizar un mínimo de comprensión del idioma como

para poder recrear un 'retrato robot' fiable.

5.4. Concepto de las Actividades

Cuando planteamos que la aplicación fuera un reto colaborativo entre alumnos, pensamos en que teníamos que garantizar un mínimo de vocabulario para poder acceder a ese reto. Generalmente en los juegos a los que jugamos hoy día se premia al usuario si ha superado un reto. Conforme se van superando esos retos, se debe aumentar sensiblemente la dificultad para exigirle más al jugador.

Con el aprendizaje es exactamente igual, no podíamos exigir menos con cada actividad de manera que cada actividad es más exigente que la anterior y contiene en su medida un vocabulario más completo que su inmediatamente anterior. De esta manera garantizábamos el continuo aprendizaje.

Pero nos enfrentábamos a un problema. El alumno podía dejar el móvil en la mesa y consultar internet desde otra vía como un ordenador y preguntar por la pregunta que se le estaba haciendo en el cuestionario. Para evitar que el alumno use otras vías salvo su habilidad con el idioma, añadimos el factor del **tiempo**. El tiempo ajustado a la dificultad de la actividad, obliga al alumno a contestar rápidamente las preguntas ya que si se detuviese a consultarlas se acabaría el tiempo y no podría puntuar más alto.

Teníamos resuelto el problema de la curva de dificultad, y teníamos resuelto el problema de que pudiesen obtener las respuestas por otro lado. Es decir que el alumno iba a resolver las actividades con sus propios medios. Pero nos dimos cuenta de que obligar al alumno a resolver en la primera vez toda la batería de preguntas que lleva una actividad podría ser cansino para tratarse de algo que debería resultar cómodo a la par que fuese un reto.

Dado que una actividad puede tener una batería de preguntas de unas 20 respuestas, y habíamos puesto que cada actividad abarcara solo 8 preguntas, teníamos que hacer que el alumno repitiera varias veces el mismo ejercicio para poder ver la totalidad del vocabulario al menos una vez. Para ello recurrimos al concepto de esfuerzo-premio. Para poder avanzar en la aplicación de actividad en actividad hasta llegar al chat, se debe superar las actividades no una, si no varias veces dependiendo de su complejidad.

Por último, teníamos que garantizar el correcto aprendizaje del alumno, no era suficiente con que el alumno repitiera x veces una actividad. Por ello exigimos un mínimo de puntuación a superar. Si se supera ese mínimo, en el límite de tiempo establecido, se considera superada la actividad y por tanto nos quedará menos para desbloquear la siguiente.

Una vez superado el conjunto de actividades el alumno debería ser capaz de desarrollar una actividad conjunta en la que el vocabulario que se usa es el mismo que el de las actividades.

Figura 5.7: Diseño de presentación de una actividad

5.4.1. Diseño de las Actividades

Las actividades están pensadas de manera que el usuario no crea que está jugando a un juego totalmente diferente y se acabe distrayendo del verdadero objetivo de la aplicación, evaluarse a sí mismos. La aplicación en sí misma es un conjunto de actividades en los que probar la destreza el usuario con el idioma.

Fue muy importante crear las diferentes actividades con el mismo modelo para que el alumno o estudiante no se sintiese bombardeado con información no relevante para su aprendizaje. Al jugador en principio no le interesa memorizar complejos modos de niveles como en los juegos comerciales, en un 'trainer' lo interesante es medir su capacidad con sencillos retos.

Las actividades se divide en tres partes:

- Introducción: (Fig. 5.7) Se nos presenta el título de la actividad en concreto y se nos detalla un pequeño texto a modo de introducción. Si pulsamos en empezar pasaremos a la segunda parte.

- Desarrollo: (Fig. 5.8) En este apartado se le presentará al alumno el reto propio de cada actividad. En todos los casos consistirá en mostrar una serie de fotos relacionadas con el vocabulario a repasar las cuales el alumno deberá interpretar para poder resolver cada pregunta. A menudo, se le presentarán varias opciones inválidas y sólo una válida. Cada opción válida elegida puntúa un punto, y cada opción inválida asumirá un punto negativo.

  El alumno deberá responder correctamente a cada pregunta de las ocho que suponen el cuestionario para poder pasar a la siguiente pregunta. En la parte superior de cada actividad tendrá una breve descripción de lo que tiene que hacer (por supuesto en el idioma extranjero) junto a un botón de 'play' que reproducirá exactamente la misma explicación

Figura 5.8: Diseño del desarrollo de una actividad

pero en sonido, para que el jugador pueda relacionar la información de texto en voz también.

Justo más abajo de la descripción, tiene un contador de tiempo que le avisará del tiempo **restante** que tiene para resolver el cuestionario **completo**. Quiere decir que tenemos un tiempo estipulado para resolver las ocho preguntas, pero no por ello es necesario contestar a todas (aunque si a casi todas), será suficiente con haber obtenido la puntuación necesaria.

El jugador dispondrá de dos métodos para resolver las actividades. Uno será pulsando botones situados específicamente para cada prueba, pero también se le exigirá la entrada de texto escrito por él mismo mediante el teclado. Esto dependerá de la actividad que estemos resolviendo en ese momento.

– Resultados: (Fig. 5.9) Al terminar el tiempo, o las ocho preguntas propuestas por cada ejercicio, se le presentará al alumno una última vista antes de volver al menú o a repetir el ejercicio. En esta vista tendremos un resumen de las respuestas que nos ha proporcionado durante la actividad.

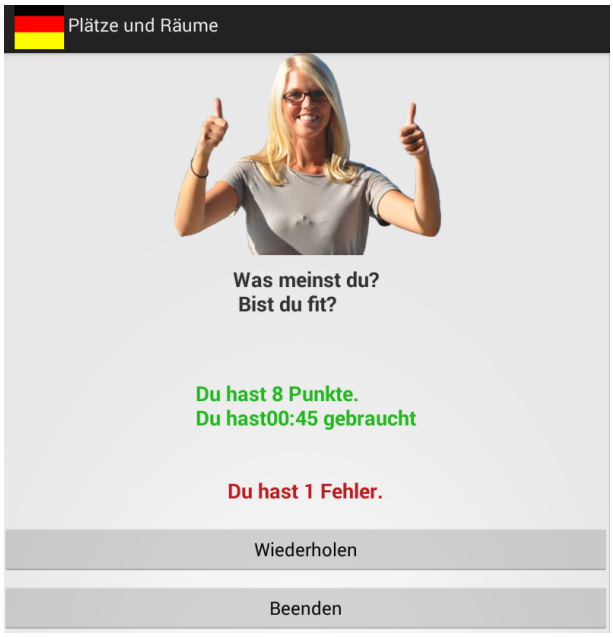


Figura 5.9: Diseño del resultado de una actividad

Es importante prestar atención a esta vista ya que en ella se nos notificará cuantas veces debemos de seguir intentándolo para pasar a la actividad siguiente, o en su defecto, que elijamos la actividad que hayamos desbloqueado. Los puntos que hayamos sacado durante la actividad se mostrarán en un mensaje en verde, y como máximo podremos sacar 8 puntos en cada intento. Por otra parte los puntos negativos o errores vendrán escritos en rojo, indicando el número de veces que hemos seleccionado una opción incorrecta.

El tiempo aparecerá en verde, y nos mostrará el en formato MM:SS el tiempo que nos ha llevado resolver el ejercicio, o si directamente no llegamos a terminarla, el tiempo que ha llegado a durar la partida. No existen hándicaps de tiempo salvo el de tener que resolverla **antes** de que se acabe el tiempo.

El resumen completo obtenido de la última actividad aprobada (o la última suspensa si aún no hemos superado las necesarias) se podrá comprobar en el apartado Punktezahl desde el menú principal

5.4.2. Encuentra el objeto

Encuentra el objeto (Fig. 5.8) es el primer modo que el alumno deberá superar. En este modo se hace incapie en la relación vocabulario-imagen. Queríamos que el alumno fuera capaz entrenando su habilidad con el idioma mediante la aplicación de relación rápidamente una imagen de un objeto con su nombre en Alemán.

Se le mostrarán cuatro imágenes, con cuatro botones situados justo debajo de ellas. Cada botón pertenece a la foto que se sitúa justo encima de ellos. Se le formula una pregunta del estilo, ¿Cual de las siguientes fotos es el hotel?. El jugador tendrá que visualizar las cuatro imágenes que se les proporciona y deberá distinguir entre ellas el objeto que se el ha pedido que distinga. Si quiere escuchar la frase en el idioma original puede hacerlo mediante el botón de play.

El tiempo indicado para este ejercicio es de **1 minuto**, hemos dejado el tiempo suficiente para que el alumno elija rápidamente entre una de las cuatro opciones, ya que si poníamos más tiempo podría no suponer un reto o que el jugador decidiera obtener la solución por otros medios. Con esa cantidad de tiempo se refuerza el conocimiento del alumno ya que no deja tiempo para pensar tranquilamente la opción si no que debe dar una respuesta intuitiva.

El hándicap de éste tipo de actividad será de 7 puntos. Para obtener ese hándicap podríamos sacar 7 puntos y 0 fallos, o 8 puntos y un fallo como máximo. Con esas puntuaciones obtendremos un aprobado en el intento que acabamos de realizar.

5.4.3. Foto escondida

Foto escondida (Fig. 5.10) es una actividad que supone una evolución de la anterior. En el anterior queríamos conseguir la rapidez de respuesta del alumno ante cuatro imágenes que no tenían que estar relacionadas entre sí. En este vamos a probar su capacidad de distinción entre imágenes y de razonamiento.

En la foto de ejemplo (Fig. 5.10) se muestran tres imágenes y el hueco de una que ha desaparecido, pero tendrá a su disposición cuatro botones de los que seleccionar el objeto que falta. Pongamos un ejemplo visual, en la foto vemos tres imágenes que son: una panadería, un jardín y un parque, y cuatro botones que son Hotel, panadería, parque y jardín. Debemos elegir aquel botón cuya imagen falte en pantalla en el caso de ejemplo si pulsamos sobre el botón 'Hotel' obtendremos un punto, y en el caso contrario obtendremos un fallo.

Una vez seleccionado la correcta, se nos mostrará otra batería de fotos hasta resolver las ocho preguntas. El énfasis de esta actividad es que el alumno sepa distinguir cuales son las tres imágenes que hay para poder adivinar la cuarta, o reconocer que foto falta leyendo los cuatro objetos en forma de botón.

Dado que esta actividad puede resultar un poco más difícil que la anterior, se ha subido el límite de tiempo a **1 minuto 10 segundos** para poder contrarestar la diferencia de tiempo que nos llevará razonar que foto falta.

La puntuación es exactamente igual que la anterior, será necesario puntuar 7 puntos para poder tener aprobado un intento de tres.

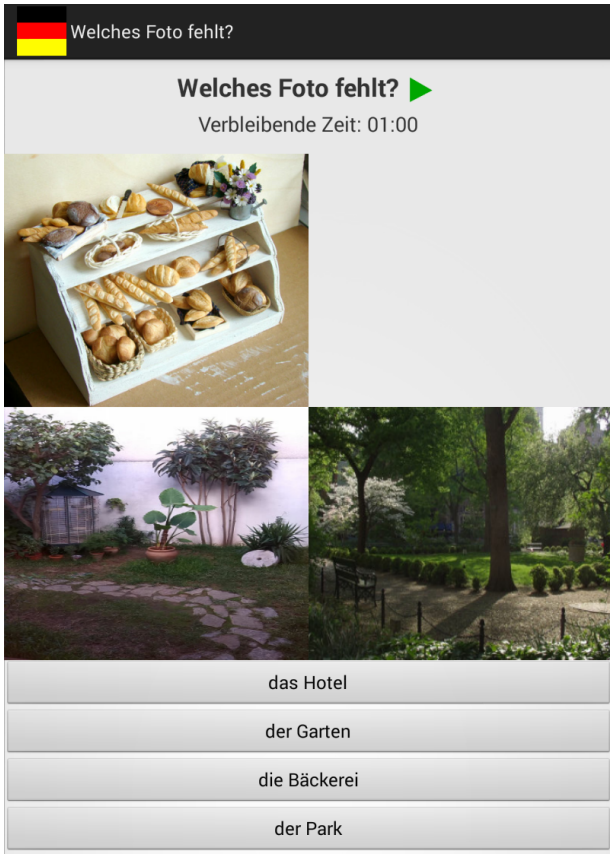


Figura 5.10: Diseño de la actividad: Foto escondida

Figura 5.11: Diseño de la actividad: Rellena los huecos

5.4.4. Rellena los huecos

Rellena los huecos (Fig. 5.11) es la última actividad planteada para el alumno. El jugador deberá interpretar que ve en la foto y rellenar los huecos que aparecen en la frase para poder completar una pregunta.

En la foto de muestra, vemos que en la parte superior se proporciona la pregunta, ¿que hace la persona y donde?. En la foto podemos ver que hay una chica leyendo el periódico. Traducido del Alemán, vemos que tenemos algo así como Ella ’ ’ Periódico en el ’ ’. El cometido del alumno es escribir en esos dos huecos diferenciados por ’ ’ la palabra que crea correcta para acertar la pregunta.

Una vez completada la frase, el jugador deberá pulsar la opción enviar para que el juego compruebe su frase. Si la frase introducida es incorrecta se avisará al jugador dos veces (con su penalización de dos puntos por cada palabra mal) y si sigue sin corregir el texto aparecerá un mensaje comentando la solución.

Tomamos ésta medida para poder proporcionar un conocimiento al alumno que puede que no tenga previamente, es decir, no nos sirve de nada dejar estancado al alumno en una pregunta si no sabe decir ’lee’ en Alemán, pues esa información se la proporcionaremos si ha fallado tres veces para que pueda continuar haciendo el ejercicio, aunque éste ya esté suspenso.

La actividad muestra una clara dificultad frente a las dos anteriores ya que ahora es el alumno quien proporciona una información que no se muestra en pantalla, aparte de que debe teclear el texto correctamente. Por ello y por la dificultad que requiere y bajo las pruebas que se han realizado decidimos dejar el tiempo en **3 minutos**.

El listón de puntos también bajó para poder adaptarse a la nueva dificultad. Esta vez permitiremos que el balance total sea de 6 puntos. Es decir que podremos sacar 8 puntos y hasta 2 fallos, 7 puntos y hasta un fallo, o 6 puntos y ningún fallo. Aparte y como nota importante ésta actividad sólo requiere ser aprobada dos veces en vez de tres. Esto se debe a que nos llevará el triple (de ahí su nuevo límite de tiempo) completarla que las otras dos, y así no fatigamos al alumno obligándole a estar tanto tiempo con una misma actividad pudiendo provocar su desinterés.

5.5. Actividad colaborativa: Catch me if you can

5.5.1. Introducción

Catch me if you can es la actividad cooperativa entre los alumnos que supone el gran reto de la aplicación. Dado que es más laborioso de explicar le he dedicado una sección entera.

En el chat los alumnos podrán intercambiar información entre ellos de manera anónima para poder resolver una actividad planteada. Habrá dos roles que se podrán interpretar durante la 'partida'. Al igual que con las actividades el chat contará con un medidor de tiempo que establecerá cuanto tiempo tienen los jugadores para resolver la actividad antes de que se cierre el chat.

Las conversaciones de los alumnos quedarán grabadas en las bases de datos y podrán ser exportadas en pdf pudiendo ver que escribió cada alumno (aunque ellos no supieran con quién estaban hablando) y poder evaluar el nivel de partición en la actividad. Queríamos evaluar el nivel de habilidad cooperativa que tenían los alumnos con el idioma. Esto hace que el flujo de información sea circular entre ellos y no sea un profesor el que tenga que estar proporcionando o corrigiendo al alumno.

5.5.2. Roles

La actividad plantea el reto de intentar evitar un asesinato ficticio (parecido a algunos juegos de mesa) mediante el uso de pistas e indicios. Los alumnos deberán de pasarse información entre ellos para que puedan ir escaneando códigos que implican más información hasta poder llegar a descubrir cual de los asesinos posibles es el correcto.

Para ello deben de coordinarse y asumirán unos roles específicos:

- Comisario: Será el encargado de leer los distintos indicios de dónde se encuentran los testigos. En el apartado del chat llamado Bloc tendrá una serie de indicios (por ejemplo, Indicio 1: El testigo está en el parque escuchando música) que deberá compartir mediante chat con sus dos compañeros. Del comisario depende la correcta interpretación de los

indicios y de su correcta transmisión a los compañeros.

También dispondrá de un botón Scan, el cual servirá para poder escanear el código (situado en las fotos de los asesinos posibles) del asesino correcto para poder resolver la actividad.

– Policía: Puede haber hasta dos policías en la partida, y serán los encargados de desplazarse por el escenario de juego (por ejemplo, la clase de la facultad, el patio...) buscando los indicios que el comisario les ha transmitido por chat. Si el indicio era El testigo está en el parque escuchando música deberán buscar la foto de un parque, y de las fotos de parques deberán encontrar aquella en la que aparezca una foto de un testigo escuchando música.

  Una vez encontrado la foto del testigo, deberán hacer uso del botón Scan para escanear con la cámara de su móvil al testigo indicado. Si escanean un código del juego, podrán ver un vídeo de un testigo diciendo una información relevante (si han escaneado el código correcto) como podría ser El asesino tiene el pelo rubio. Esa información deberán de transmitirla por chat de nuevo al comisario, para que éste pueda seguir proporcionando más indicios de los testigos.

5.5.3. Explicación

Como ya hemos descrito, ésta actividad chat supone el esfuerzo de tres jugadores trabajando juntos para resolver un reto. Quién inicie primero el chat de un grupo de tres o dos alumnos será quién asuma el control del comisario. Los comisarios estarán situados en una zona con varios carteles de fotos con asesinos debidamente descritos.

Se podrán observar los gustos del asesino, por ejemplo si escucha música o si lleva pulsera. Entre los asesinos podrá haber ligeras diferencias, ya que los diseños que hemos propuesto son las mismas personas con ligeras modificaciones para que no se sepa fácilmente cual es cual. El comisario deberá saber distinguir al verdadero de entre los falsos.

Al principio solo tendrá a su disposición dos indicios, que deberá asignar a sus dos compañeros policías para que vayan a escanear los códigos pertenecientes a tales testigos. Una vez que sus compañeros hayan escaneado al menos uno de los dos indicios, el comisario tendrá desbloqueado dos más para poder transmitir a sus compañeros. Está pensado de manera que si un policía no consigue acertar con su testigo el juego pueda seguir adelante.

Cada testigo escaneado supone una pista en forma de vídeo y por tanto información relevante que deberá llegar hasta el comisario, quién debe apuntarlo en el apartado de su Blog. Conforme vayan avanzando en el juego el comisario reunirá pistas suficientes como para poder distinguir cual de los asesinos mostrados es el correcto. Una vez que lo haya distinguido deberá escanear

el código del asesino y obtendrá una confirmación de que lo ha encontrado o una negación.

Una vez terminado la actividad, se mostrará en el chat las pistas que ha ido analizando el comisario para que pueda ser analizado por el profesor su trabajo en grupo.

5.6. Diseño de clases

En esta sección vamos a hablar del diseño de las clases y de sus características.

5.6.1. ChatMain

Empecemos por la más importante, es la encarga de gestionar el estado actual de nuestra conversación del chat. Depende de otras clases que implementan los fragmentos de las diferentes vistas de la conversación chat, como son *ChatConversationFragment, AssignmentFragment y VideoPlayerFragment*.

La clase **ChatMain** tiene como atributos importantes los siguientes:

- **muc** variable que contiene los datos necesarios para la comunicación con la conversación múltiple.
- **Roomname** nombre de la sala en la que estamos, cuando se produzca un envío de mensaje se enviará a la sala descrita por esta variable.
- **timer** cronómetro que controlará el tiempo límite que tiene el comisario para resolver el juego junto a sus compañeros.
- **offline_messages** cola de mensajes offline que no han sido enviados por una desconexión con el servidor.
- **begin** fecha que indica el comienzo de la partida
- **end** fecha que indica el final de la partida

Los métodos que implementa, son necesarios para que el juego funcione correctamente mencionaremos los más importantes ya que algunos son más técnicos y no tan representativos:

- **onSendClick** se encarga de enviar el mensaje que el usuario haya escrito a la conversación adecuada, además de añadir el momento de tiempo en el que fue enviada.
- **onBlocClick** este método se encarga de cargar la vista del blog y mostrar al comisario su blog de notas.

- **onScanClick** método que se encarga de llamar a la aplicación BarcodeScanner
- **notifyTimer** este método se ejecuta automáticamente cuando se acaba el tiempo. Si no se ha ganado con anterioridad cargará la derrota del jugador.

Como se puede apreciar, esta clase es la encargada de mantener todos los elementos del juego colaborativo funcionando en común, pero necesita de otra clase que haga de capa intermedia entre el protocolo XMPP y nuestra clase principal. Esa clase es la clase ChatHelper.

5.6.2. ChatHelper

Ésta clase fue necesaria de desarrollar debido a la cantidad de métodos que implementaba el protocolo XMPP. Para hacerme más sencillo el desarrollo de la aplicación, hice esta clase para encapsular todo lo referente a la comunicación con la librería Smack. Su único atributo público es el atributo connection, que almacenará el estado de la conexión con el servidor.

Sus métodos importantes son los siguientes:

- **connect** El método mediante el cual comienza a funcionar la comunicación, necesitaremos un nombre de servidor y un puerto por el que conectarnos. Nos devolverá true si la conexión ha sido correcta y false en caso contrario.
- **disconnect** Nos desconecta del servidor del que estemos conectados.
- **login** Hará que iniciemos sesión al servidor que le hemos proporcionado por el puerto proporcionado. Devuelve true si el usuario y contraseña corresponden al login original y si se ha podido establecer el inicio de sesión. Necesario si queremos enviar mensajes.
- **sendGroupMessage** Enviará un mensaje a la sala que le describamos, con el contenido del mensaje que le proporcionemos. Mientras no lleguen a salir del dispositivo almacenará los mensajes en una cola.

Sin esta clase, aunque tengamos desarrollado el funcionamiento de la clase principal del chat, no tendríamos comunicación con el servidor, ya que se encarga plenamente de las comunicaciones con éste.

5.6.3. DatabaseHelper

DatabaseHelper es la clase que hemos pensado para guardar las partidas en el dispositivo del usuario. Como hemos utilizado SQLite3 para almacenar y gestionar las bases de datos, vi necesario utilizar una clase que fuera capaz de abstraer el uso de la base de datos.

Los métodos más importantes encargados de la gestión de la base de datos son los siguientes:

- **loadSaveGame** Método que carga la partida guardada a la memoria principal del dispositivo. Devuelve el valor de la actividad más alta a la que hemos llegado, para así poder continuar por dónde lo dejamos.

- **checkRemaining** Este método devuelve el número de veces que debemos superar una actividad para poder desbloquear la siguiente.

- **saveToDB** Necesario para guardar el número de aciertos, de fallos, el tiempo necesario, y cuando ocurrió, la prueba de una actividad.

5.6.4. Las clases SearchScene, HiddenPicture y FillTheGasp

Estas tres clases son las encargadas de manejar las actividades que tienen el juego. Tienen muchos aspectos en común y por ello he decidido ponerlas juntas.

Las tres clases reciben un entero por parámetro que les indica con que temario estamos jugando. Esto quiere decir que las tres actividades tienen tres temarios distintos disponibles y dependiendo de uno o de otro cargará unas preguntas u otras.

De igual manera, utilizan un método **notifiyTimer** para reconocer cuando se ha acabado la actividad, y entonces mostrar un resumen de como se ha jugado utilizando el método **loadResult**. Este método mostrará tanto fallos como aciertos como el tiempo que ha llevado al jugador jugar la actividad.

También dependerán de la clase **Mark** ya que es la que se encarga de anotar los aciertos, fallos, e incluso las frases que nos va a aconsejar al final de la partida cuando se cargue el resultado.

Para ayudar al usuario, tienen un método que reproducirá la pregunta explicativa de la actividad, llamado **onQuestionSoundClick** y en el caso de las dos primeras clases, también tendremos un método de ayuda que reproducirá un sonido diciendo la foto que hemos escogido, llamado **startPhotoSound**.

La manera de brindar las diferentes opciones al usuario, es la misma. Para iniciar una partida, se hará a través del método **onAcceptClick** que se encargará de llamar al método **reloadActivity** que también servirá tanto para cargar la actividad por primera vez como la n vez. Para repetir, se ejecutará el método **onRetryClick** que llamará de nuevo a la función reloadActivity, en cambio si el usuario pulsa salir las clases tienen un método **onExitClick** que cerrará la actividad y cargará el menú principal.

Como diferencias, la tercera clase analizará el resultado de una respuesta cuando el método **onSendClick** se ejecute, ya que éste es el encargado de tomar la frase aportada por el usuario y valorarla. En cambio las dos primeras tendrán un método por cada foto llamado de la forma **onSelectXClick** donde *X* es el número de la foto.

Después de haber destacado los parecidos y las diferencias de las tres clases encargadas de las actividades, podemos decir que son prácticamente idénticas salvo a la hora de recoger la entrada del usuario, ya que la manera de jugar es ligeramente distinta pero se mantiene la misma estructura de juego. En todos los casos, se depende de la clase DatabaseHelper para poder interactuar con la base de datos a la hora de guardar la partida.

5.7. Aspectos técnicos del chat

5.7.1. Login

En la vista de login se nos permitirá poder inciar sesión en el servicio de chat de openfire. Para poder iniciar sesión deberemos proporcionar nuestro usuario y contraseña perteneciente a nuestra cuenta en el servidor. Por lo tanto debemos estar registrados con anterioridad antes de poder acceder al chat. De ninguna manera la aplicación permitirá gestionar nuestro usuario. Es decir que no podremos ni registrarmos ni borrar nuestra cuenta del servidor.

Tampoco podremos gestionar nuestra lista de amigos ni podremos agregar a nadie, ya que es el sistema el que tiene organizado a cada cuenta con su grupo de amigos para poder formar los grupos posteriormente en el juego.

5.7.2. Empezar partida

Para empezar la partida, será necesario que un jugador de cada grupo pulse la opción de comenzar partida una vez iniciado sesión. El primer usuario de cada grupo en pulsar comenzar partida creará automáticamente una sala en la que se desarrollará la partida con el nombre testx (donde la x es el número de la sala).

El creador de la sala asumirá el rol de comisario. Esto se debe a que hay ciertas variables que guarda solo un jugador para poder establecer los parámetros del juego como las pistas o los indicios. Por tanto la partida dependerá casi al completo de él.

5.7.3. La mensajería

Una vez iniciado sesión y estando en la vista de la conversación, podremos enviar y recibir mensajes de nuestros compañeros. Los mensajes vendrán diferenciados por colores para poder facilitar la comprensión de los mismos ya que no sabemos realmente con quién estamos hablando. Cada usuario será un color designado por la aplicación.

En cambio los mensajes de aceptación y rechazo en respuesta al escaneo de un código serán de color verde y rojo respectivamente. Lo hemos pensado así ya que había que diferenciar entre texto escrito por los usuarios y texto aceptado por el sistema del juego. Si escaneamos una foto

y es la foto incorrecta veremos un mensaje diciendo que ese no es el indicio correcto en rojo.

Aparte, el diseño de fondo de la ventana de la conversación es azul para poder distinguir a golpe de vista en que apartado estamos ahora mismo, ya que notamos que podía ser un gran lío mental saber si estamos en el apartado Blog o conversación.

5.7.4. Códigos y botón Scan

Cuando recibamos una información de nuestros compañeros, a menudo deberemos escanear un código. Para ello tendremos que hacer uso del botón Scan situado en la parte superior de la vista de la conversación. Una vez pulsado el botón Scan la aplicación llamará a Barcode Scanner dándole la orden de escanear un código.

Bastará con acercar nuestra cámara a un código y tomar un buen ángulo del código para que el código quede escaneado y el juego nos muestre la información que andamos buscando.

Si escaneamos un código que no tenga nada que ver con nuestra aplicación el programa nos notificará de que el código es incorrecto, pero no nos penalizará de ninguna manera ya que pueden existir malentendidos con los códigos y no se reconocen a simple vista.

5.7.5. Blog de notas

En esta vista tendremos una lista con el conjunto de indicios a comunicar a nuestros compañeros (si somos comisarios no aparece este botón). Se irán añadiendo indicios de dos en dos conforme los compañeros vayan resolviéndolos y vayan comunicando su información al comisario.

Más abajo tendrán cuatro apartados en los que podrán apuntar las características descritas por sus compañeros del asesino que intentan descubrir. Es de vital importancia rellenar esos campos pues serán supervisados por el profesor una vez terminada la actividad.

Una vez hayamos terminado de revisar los indicios y de apuntar nuestras pistas, podemos volver pulsando atrás o usando el botón añadido en la imagen para poder regresar a la conversación, ya que los cambios se quedan guardados automáticamente.

Esta vista tiene un diseño amarillo para diferenciarse de la conversación, intentando imitar al color de un bloc de notas.

Capítulo 6

Implementacion

6.1. Introducción

En este capítulo vamos a comentar todos los problemas que nos hemos encontrado a la hora de implementar el diseño del proyecto. El proyecto no ha estado exento de problemas en su implementación y más bien diría que nos hemos encontrado con bastantes. No es de extrañar ya que hemos querido mezclar tecnologías que a priori, no están muy relacionadas.

Aunque el protocolo XMPP lleva ya tiempo funcionando en ordenadores de escritorio, en dispositivos Android no hay tanta documentación ni ejemplos ya que no ha pasado tanto tiempo desde el auge de Android comparado con el de las tecnologías chat. Nos hemos encontrado problemas de todo tipo en este apartado y ha sido el más difícil de implementar.

Aparte del grandísimo esfuerzo de todo el material que incluye la aplicación (imágenes, vídeos, sonidos) hemos tenido siempre presente que el tamaño de la aplicación podía limitar la cantidad de usuarios que la utilizaran. Hemos tenido que hacer muchos cambios y adaptar el material que queríamos mostrar para que la aplicación tuviera un tamaño aceptable para los usuarios pero que no se viera afectada su utilidad.

6.2. Android y el manejo de la clase Activity

Para desarrollar una aplicación en Android, se deben tener en cuenta ciertos aspectos que se diferencian de desarrollar en otros sistemas. En la propia API de Android podemos encontrar varias lecciones sobre cómo desarrollar para este sistema. A la hora de desarrollar el proyecto, me encontré con varias dificultades a la hora de implementarlo para Android, esas dificultades están relacionadas con el término de Activity y del tiempo de vida de éstas.

Como en la propia API indica, las activitys son las clases que se encargan de la lógica y de la correcta visualización de una tarea de la aplicación. Por ejemplo el menú es una activity y el apartado puntuaciones es otra activity. Cada activity tendrá su conjunto de variables y de métodos que conformaran la lógica de la propia ventana que estemos mostrando. En la figura 6.1

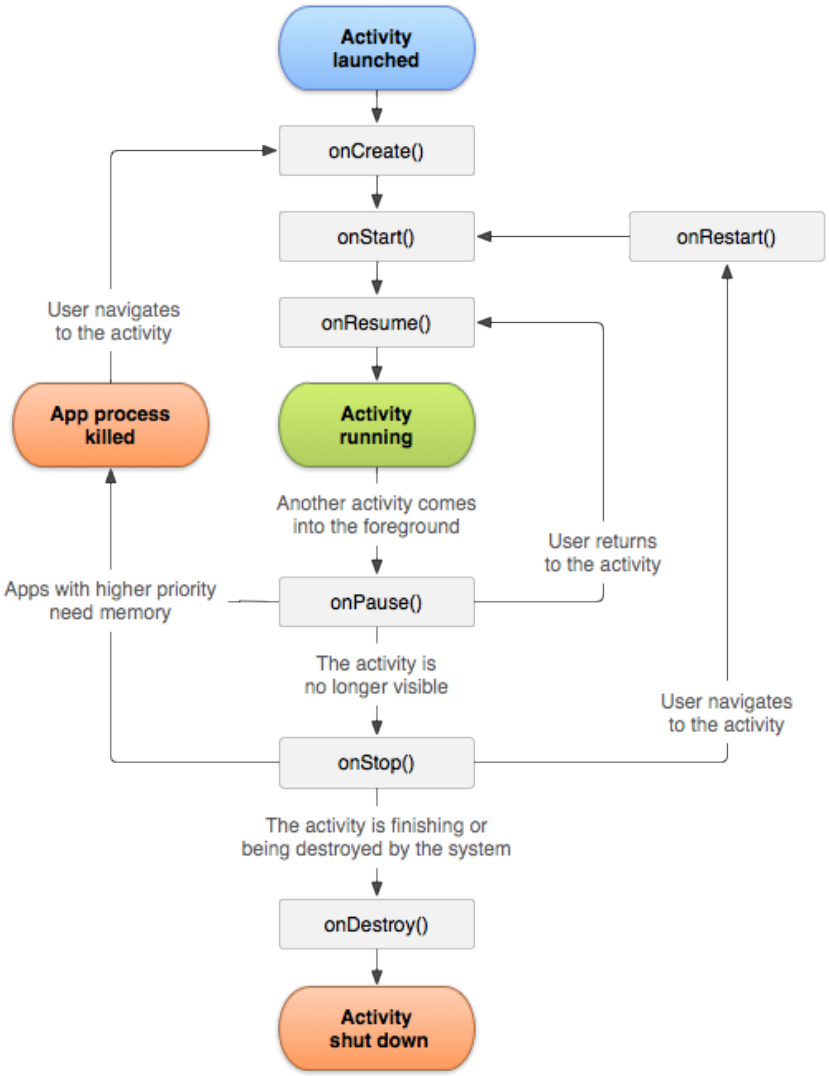


Figura 6.1: Diagrama del ciclo de vida de una Activity

podemos ver cómo es su ciclo de vida.

Cada vez que creamos una activity, se llama al método onCreate() que se encargará de inicializar los valores de las variables del objeto que residirá en memoria. También tendremos métodos que serán escuchadores que reaccionarán a las pulsaciones sobre los botones de la pantalla. Hasta ese punto todo está bien estructurado y no presenta grandes complicaciones.
En cambio, ¿Qué ocurre cuando giramos la pantalla? Pues que desgraciadamente la activity se crea de cero de nuevo, se llama obligatoriamente a onCreate. Esto es un inconveniente ya que si giramos la pantalla de la Tablet perdemos todos los cambios realizados en el objeto que estaba ejecutándose en memoria. La solución pasa por utilizar en el método onCreate el parámetro que nos viene de entrada llamado savedInstanceState.

Con éste parámetro, podremos diferenciar si la activity ha sido creada por primera vez o si ha sido creada forzosamente después de un giro de la pantalla. Otra opción es utilizar el método onRestoreInstanceState pero no es la opción que hemos utilizado. Cada vez que se llame al método onCreate hacemos la comprobación de que savedInstanceState no es null, para recuperar los datos almacenados en savedInstanceState y recrear la aplicación tal y como se dejó antes del giro. Para guardar información antes del giro de pantalla debemos sobrecargar el método onSaveInstanceState en el que introduciremos los valores de las variables tal y como están actualmente para poder recrearlo posteriormente.

En principio esta herramienta parece ser suficiente para poder mantener la ilusión de que la activity siempre está ejecutándose y no se ha producido ningún corte en su ejecución, aunque realmente si haya ocurrido y ahora tengamos un objeto con el mismo estado que el objeto que acabamos de destruir al girar la pantalla. Pero entonces me encontré con otro problema, sólo podemos guardar objetos de los tipos más comunes, como integer, string... y conjuntos como arraylist pero de los mismos tipos.

Como opción alternativa, Android propone que si tienes un objeto que quieres guardar y no es del tipo admitido, entonces el objeto que quieras guardar debe implemente la clase Parcelable de Android, o Serializable de Java. Esto se debe a que debe tener ciertos requisitos para poder ser guardado en un objeto Bundle (como el conocido savedInstanceState). Es decir que si creamos una clase que queremos guardar en algún momento en un Bundle necesitamos que la clase implemente la clase Parcelable y sobrecargue sus métodos. Pero eso tampoco solucionó el gran problema que teníamos, el de querer guardar un objeto XMPP connection.

6.3. Conectividad XMPP a lo largo del chat

XMPP connection es una clase que nos proporciona conectividad a un servidor con el protocolo XMPP a nuestro programa, para cualquier comunicación que queramos establecer con el servidor necesitaremos hacerlo mediante ese objeto, por lo que era vital para el funcionamiento del proyecto. Y ahí residía el gran problema, si girábamos la pantalla, el objeto connection que nos mantenía conectados a la red de chat se destruía y por desgracia, no podíamos sobrecargar

la implementación de la clase para convertirlo en Parcelable o Serializable.

Este problema fue quizás el que más tiempo me llevó solucionar y no encontrábamos mucha documentación al respecto, salvo lo que hemos mencionado con anterioridad. En algunas aplicaciones que podemos descargar del mercado de aplicaciones de Android, podemos observar que no permite el giro de pantalla. Esto puede ser debido a las mismas razones técnicas que hemos comentado.

La solución que tomé fue la de investigar si existía algún objeto que se mantuviese **siempre** ejecutándose mientras se ejecutaba la aplicación. Ciertamente Android tiene esa clase y se llama Application. Bastó con crear una nueva clase llamada PFC que heredase de la clase Application e ir almacenando en ese objeto los objetos de **vital** importancia para la aplicación, aquellos que no podían implementar la clase Parcelable debido a su naturaleza.

Se podría decir que mientras que la activity que se encarga de la lógica del chat se destruye y se recrea cada vez que se gira la pantalla en la conversación, el objeto pfc siempre está vivo albergando el objeto xmpp connection que mantiene la conexión viva entre la aplicación y el servidor.

Esto no solo nos permitía mantener viva la conexión cuando se producía un giro de pantalla, si no que también servía para cuando cambiábamos entre las dos ventanas principales del chat, la vista del login y el chat principal, no se perdiese la conexión que ya teníamos.

6.4. Cronómetro

En los anteriores apartados explicamos cómo se crea y se destruye una actividad de Android y que ocurre cuando hacemos un giro de pantalla. A la hora de querer crear un cronómetro tuve que tener en cuenta el ciclo de vida y el conjunto de estados de una actividad. No sólo por el giro de pantalla en el que se crea y se destruye la activity, si no también en los estados por los que pasa cuando minimizamos la aplicación o pasa a segundo plano por otra.

Para entender esto, es importante mirar la figura 6.1 de nuevo, en ella podemos ver los diferentes estados por los que pasa una activity. Si la activity pasa al fondo por que haya salido un diálogo, entonces se llamará al método onPause, y si la aplicación es minimizada se llama al método onStop.

Para recuperarse de esos dos estados, la activity llama a onResume (si estaba en pausa) y a onRestart (si estaba parada). Estos dos métodos permiten controlar que sucede en cada estado de la aplicación para poder aplicarle nuestros requisitos. Por ejemplo nos dimos cuenta de que al minimizar la aplicación, el tiempo no avanzaba. Siendo así, alguien sería capaz de leer la pregunta, minimizar la actividad, buscar la solución por internet y luego volver a la actividad para completar la pregunta que se le había hecho sin apenas gastar tiempo de cronómetro. Esto si suponía un problema grave ya que si se descubría afectaba directamente al reto de tiempo que

se supone que debe tener cada actividad.

Para evitar esos problemas, la manera que utilicé para contar el tiempo fue la siguiente:

- Implemente una clase llamada TimerClass para poder controlar el cronómetro de manera estándar. A ésta clase se le pasa un parámetro al constructor que será la fecha fin a la que saltará el cronómetro.
- Cuando empezaba una actividad, tomaba el tiempo que daba la actividad (por ejemplo, 70 segundos) y sumaba 70 segundos a la fecha actual, con lo que obtenía cuando debía saltar el cronómetro. Por ejemplo, si eran las 14.00.00, al constructor le pasaba un objeto Date que contenía la hora 14.01.10.
- Cuando se producía un giro de pantalla, guardaba el valor de la fecha de inicio y la fecha de fin, es decir guardaba dos objetos Date (begin y end) que contenían los valores 14.00.00 y 14.01.10 respectivamente.
- Al volver del giro de pantalla simplemente creaba otro objeto TimerClass (mi cronómetro) con la fecha end que había guardado en memoria y volvía a poner en funcionamiento el cronómetro.
- Cuando alguien minimizaba la aplicación, la aplicación entraba en el estado onStop y se paraba completamente el cronómetro. Esto dejaba la aplicación en suspensión por así decirlo.
- Al regresar de un estado de pausa en el que el cronómetro había sido invocado por otro hilo de ejecución no era capaz de hacer cambios en la vista de la Activity, por lo que había que crearlo de nuevo. La solución fue en el método onRestart parar el cronómetro que teníamos ejecutándose y crear uno nuevo a partir de la fecha fin que teníamos.

De esta manera, conseguimos evitar que los usuarios pudieran alterar el correcto funcionamiento del cronómetro durante las actividades de la aplicación.

6.5. Javadoc

Para entender debidamente un código, éste debe de tener comentarios que expliquen su funcionamiento a aquel que lo desee leer. Pero a veces con eso no es suficiente, se debe seguir unos requisitos a la hora de comentar correctamente un código.

Aunque dentro de cada método expliquemos la importancia de ciertas instrucciones y de que hacen éstas, también debemos de seguir un orden explicando sus precondiciones y poscondiciones. También debemos de explicar brevemente que hace cada clase para que el desarrollador

que esté interesado en utilizar nuestro programa sepa de que se encarga cada módulo del programa.

Para ello utilicé Javadoc, ya que está desarrollado por Oracle (los encargados de desarrollar Java) y está integrado en la mayoría de los IDEs compatibles con Java, por lo que desde Eclipse fue muy sencillo generar documentación en Javadoc.

6.6. Multilenguaje

Aunque nuestra aplicación fuese enfocada al idioma Alemán, queríamos que fuera totalmente compatible con otros idiomas. Ante la ignorancia pensaba que habría que diseñar otra vez toda la interfaz a la hora cambiar un idioma, o que habría que hacer cambios mayores. En ese sentido Android facilitó bastante la implementación.

En la versión final del proyecto sólo incluye el idioma Alemán, pero en las pruebas de concepto probamos a usar tres idiomas diferentes que eran el Español, Aleman e Inglés. Para ello, Android recomienda como mejor opción definir los textos dentro de un fichero xml del directorio de recursos de la aplicación.

Primero vamos a explicar cómo toma los recursos dependiendo del idioma y luego cómo se define un recurso para un idioma:

Definiendo un idioma: Android sugiere que coloquemos nuestros ficheros de recursos en la carpeta asociada a su idioma. La carpeta de recursos se encuentra en el directorio res (de *resources* en inglés) y dentro contendrá los diferentes recursos de la aplicación. Si nuestra aplicación tiene dos idiomas, por ejemplo el Español y el Francés, tendremos que crear dos carpetas una por cada idioma.

De esta manera definiremos dentro del directorio res los directorios values-fr (francés) y values-es (español). Si nuestro dispositivo está configurado con el idioma Francés cogerá los recursos de la carpeta values-fr y si está definido con el idioma Español de la carpeta values-es.

¿Pero qué ocurre cuando nuestro dispositivo está en un tercer idioma que no coincide? Aquí lo recomendado es hacer lo siguiente:

Dentro de la carpeta values (carpeta de recursos por defecto) debemos de meter los recursos que queramos que aparezcan en caso de que no esté configurado ni en un idioma ni en otro. Así nos aseguramos de que en caso de que el jugador no tenga configurado el idioma Francés o Español pueda cargar un idioma por defecto.

Siendo así nuestra aplicación no tiene ninguna carpeta de idioma alternativa, ya que queremos que sea el idioma que sea cargue los únicos recursos que tiene por defecto, es decir, el del Alemán. Pero si algún día se quisiera ampliar a Inglés solo haría falta añadir un directorio llamado

values-en que contuviese los recursos en Inglés de la aplicación.

Definiendo un recurso: Dentro de cada carpeta values que simboliza un idioma cada uno, tendremos varios ficheros xml. Por defecto nos vendrá sólo strings.xml (cadenas de texto), en este fichero podemos definir cadenas de texto generales como puede ser el título de la aplicación. Para definir un recurso se debe hacer tal que así:

- <string name=”hello”>hello</string>

De esta manera, el recurso con nombre ‘hello’ contendrá la cadena ‘hello’. Si queremos que en español ponga ‘Hola’ nos tendremos que ir a la carpeta values-es y en el fichero strings.xml añadir la siguiente línea:

- <string name=”hello”>hola</string>

Así, la traducción se hará automáticamente y no tendremos que gestionar nada más. Esta manera de trabajar nos brinda una gran ventaja, como por ejemplo poder crear varios ficheros de recursos. En el caso de mi aplicación, por cada conjunto de actividades tengo un fichero xml que contiene la definición de las cadenas.

Si queremos añadir una nueva actividad podemos crear un fichero xml nuevo y escribir sus recursos ahí, o si queremos traducir una actividad en concreto solo hace falta coger el fichero fills.xml (por ejemplo, este es el de rellenar los huecos) y traducir las cadenas de texto que aparecen en el fichero de recursos.

Con el sistema de recursos que plantea Android se puede traducir la aplicación a varios idiomas conservando su funcionalidad, sin tener que reprogramar la lógica del juego.

6.7. Reconocimiento de códigos

Para el reconocimiento de códigos, ya hemos mencionado que utilizamos la librería Zxing, pero de un modo distinto del que se espera. Una opción era incluir la librería dentro del propio proyecto e intentar solaparla con nuestro proyecto. Pero ese método nos daba bastantes problemas a la hora de ejecutar el Barcode Scanner.

Para separar nuestra implementación del proyecto de la del escáner de códigos lo más recomendado era utilizar unas clases de ayuda que proporcionaban los propios desarrolladores de la librería. Estas dos clases se llaman IntentIntegrator e IntentResult. Como es de imaginar, la primera sirve para integrar la llamada al intent de la aplicación BarcodeScanner, que nos gestionará todo el proceso por nosotros. Si la aplicación no está instalada, se le preguntará al usuario de si desea instalarla. Si responde afirmativamente se le llevará a la aplicación Barcode Scanner

en Google Play de manera que el usuario solo tendrá que pulsar en instalar, en caso contrario la aplicación seguirá su curso.

La segunda clase es la que nos proporciona el resultado del código escaneado, por lo que gracias a esas dos clases el uso de los códigos Qr fue totalmente transparente para nosotros ya que no tuvimos que adaptar la librería a nuestras necesidades.

Para invocar un Intent de escaneo, lo hacíamos creando un nuevo objeto del tipo IntentIntegrator y usando su método initiateScan. Para recibir el código escaneado, teníamos que hacerlo como en el siguiente código:

```
1
public void onActivityResult{int requestCode, int resultCode,
3                                Intent data){
    super.onActivityResult(requestCode, resultCode, data);
5
    if(data != null)
7   {
        //Rescatamos el valor del codigo
9       String codigo = data.getstringExtra("SCAN_RESULT");
    }
11 }
```

6.8. Mensajes: envío y recepción

Para el envío y recepción de mensajes nos encontramos ante otras grandes dificultades. Por un lado, había que implementar el envío del mensaje, mantener una copia de la conversación, refrescar la pantalla después de cada recepción. Para solventarlo tuve que investigar como funcionaba realmente el protocolo XMPP y entender cuales eran los pasos a seguir.

6.8.1. Envío de mensajes

Solventar el problema del envío de mensajes fue más sencillo que el de la recepción de los mismos, aun así teníamos que implementar ciertos aspectos. En el siguiente fragmento de código se pueden observar los pasos seguidos para enviar un mensaje de texto a una conversación múltiple dada:

```
1
public boolean sendGroupMessage(String text, String room) throws XMPPException{
3         Message message = new Message(room, Message.Type.groupchat);
          message.setBody(text);
5
          //Comprobamos si estamos conectados
7         if(connection.isConnected()){

9             MessageEventManager.addNotificationsRequests(message, false, true, false,
                  false);

11            //Metemos en la cola de mensajes el mensaje enviado
              messages_queue.add(message);
13
```

```
                connection.sendPacket(message);
                return true;
            }
            else{
                System.out.println("No hay conexion, no se puede enviar el mensaje");
                return false;
            }

    }
```

Como requisito y para evitar imprevistos, primero compruebo que tenga conexión al servidor, si la tengo, lo que hago es añadir una petición de notificación de envío de mensajes. Esto lo hago por que posteriormente compruebo que el mensaje haya llegado correctamente a la sala. El siguiente paso, en el que hago connection.sendPacket(message) es cuando finalmente lanzo el mensaje que quiero mandar a la conversación.

Una vez enviado un mensaje, en caso de que se ha enviado correctamente actualizo la conversación que hay en pantalla, en el caso contrario lo añado a la cola de mensajes offline :

```
//Si el mensaje no se ha enviado lo ponemos a la cola
if(sent == false)
{
    offline_messages.add(message);
    showToast("Conexion no establecida, se enviara el mensaje luego automaticamente");
}

//Si se ha enviado actualizamos el texto en pantalla
else if(onGroupChat){
    conversation_fragment.setText((String) app.getMultiChatLog());
    conversation_fragment.scrollDown();
}

conversation_fragment.getMessageText().setText("");
```

Si se muestra por pantalla el mensaje ha sido finalmente enviado, pero en el caso de que no se haya enviado por que estemos sin conexión, existe un mecanismo que salta cuando recuperamos la conexión, y que envía los mensajes que hay en cola:

```
private void enableConnectionListener(){
    app.getChatHelper().getConnection().addConnectionListener(connection_listener = new
        ConnectionListener() {

        public void reconnectionSuccessful() {
            showToast("Conexion recuperada");

            System.out.println("Reconectado");

            app.getChatHelper().getConnection().addConnectionListener(connection_listener
                );

            if(inRoom==true) {
                    onGroupChat=true;
                        inRoom=true;

                        muc = new MultiUserChat(app.getChatHelper().getConnection(), room
                            );

                try {
                    muc.join(mymucusername);
                } catch (XMPPException e) {
```

```
20                      e.printStackTrace();
                    }
22              }

24              System.out.println("Joined again to "+room);

26              //Enviamos los mensajes de la cola acumulados (no han sido confirmados sus
                    recibos aun)
                sendQueueMessages();
28
                //Enviamos los mensajes offline
30              sendOfflineMessages();

32              }
    });
34  }
```

Como se puede observar, una vez ocurrida una reconexión, procedemos a entrar otra vez en la sala y posteriormente a enviar los mensajes en cola que tengamos (aquellos de los que no hemos recibido notificación de que llegaron al destinatario, es decir la sala) y aquellos mensajes offline que tengamos acumulados (mensajes que se han escrito cuando teníamos la certeza de que no había conexión alguna).

6.8.2. Recepción de mensajes

La principal diferencia que nos encontramos a la hora de recibir mensajes frente a enviarlos, fue que teníamos que activar un mecanismo para poder reaccionar cuando nos llegue un mensaje. En algunas tecnologías móviles más antiguas teníamos que deslizar el dedo hacia abajo para refrescar la conversación. Esto puede ser un método interesante si los mensajes no fueran instantáneos, pero como se trata de un chat, teníamos que mostrar el mensaje recibido tal y como llegue a nuestra conversación.

Para ello tuve que hacer uso de unos listener (escuchadores) proporcionados por la librería asmack que me permitía mantenerme a la escucha de mensajes entrantes, y para poder refrescar la pantalla tuve que hacerlo en tiempo real con un hilo de ejecución que se encargara de realizar la tarea:

```
2   private void enableMultiChatMessageListener(){
        muc.addMessageListener(new PacketListener() {
4               public void processPacket(final Packet packet)
                {
6                   runOnUiThread(new Runnable() {
                    //necesitamos correrlo en un hilo ya que si no no se actualiza en tiempo
                        real
8                        public void run() {

10                          //Aqui va el manejo del mensaje

12                      }
                     });
14              }
    });
16  }
```

Una vez recibido un mensaje y dentro del cuerpo de la función de arriba, había que diferenciar si lo que habíamos recibido era un código o era un mensaje de texto normal y corriente:

```
1
final Message message = (Message) packet;
3 //filtramos nuestros mensajes, cogemos la parte del usuario y comprobamos si somos
      nosotros
if(message.getFrom().toString().contains("/")){
5       int index = message.getFrom().toString().indexOf('/');
        String sendername = message.getFrom().toString().substring(index+1, message.getFrom
            ().toString().length());
7
        if(!(sendername.toString().equals(mymucusername))){
9           if(message.getBody().toString().contains("c0de_")){ //Nos han enviado un codigo
                    if(host && game_over == false){//y soy el coordinador y no ha terminado
                        la partida
11                          checkQrCode(message.getBody());
                    }
13          }else{
                updateMultiChatTextValue(message);
15          }
        }
17 }
```

Si habíamos recibido un código (diferenciado por el prefijo c0de_) y eramos el que había creado la partida, teníamos que analizar el código que habíamos recibido para dar el visto bueno o dar por inválido el código recibido. Si no somos los creadores de la partida (jugamos con el rol de policía) sólo podíamos actualizar la pantalla si habíamos recibido un mensaje de texto normal y corriente.

6.9. Compresión: Audios, vídeos, imágenes

Como ya hemos explicado anteriormente, la compresión iba a ser un apartado muy importante en el proyecto. No todos los alumnos iban a estar dispuestos a instalarse una aplicación con el requisito de 20 MB de memoria libre. Por lo que nos vimos obligados a comprimir todo material que ocupara más de la cuenta.

En el apartado de **audio** fue quizá el apartado donde menos se nota la pérdida de calidad, ya que estamos hablando de que será reproducido a través del altavoz de un móvil o si el alumno así lo quisiera, a través de unos auriculares en los que iba a valorar más que se escuchase alto y sin ruidos, a la calidad misma del propio sonido.

Cuando se trata de música hay mucha información por segundo en el sonido, por lo tanto la tasa de bits (cantidad de bits por segundo que tiene el fichero de audio) debe ser alta para que se pueda distinguir correctamente el sonido de una canción. En cambio en un sonido de grabación de una profesora diciendo una frase podíamos reducir bastante la tasa de bits mientras que la voz se escuchara correctamente, ya que no iba a tener otros ruidos añadidos.

Llegamos a bajar la tasa de bits del sonido hasta los 48 Kbits por segundo, frente a los 130 Kbits por segundo que tenía la toma original desde la grabadora. Al final, la ganancia en tamaño fue la

de dos tercios del tamaño original, para aclarar, teníamos grabaciones de 34KB, que finalmente ocuparon unos 12KB.

En definitiva, logramos ocupar un tercio del tamaño que hubieramos dedicado a los sonidos de no haber usado compresión.

En cuanto al apartado de **imágenes**, hubo que tener en cuenta más detalles, aunque tomamos la misma idea, reducir los recursos hasta dejarlos a un nivel dónde no afectaran a la jugabilidad pero que ocupasen lo menos posible.

Las tomas originales de las fotos ocupaban 3MB de memoria con una resolución de 4288x2848 píxeles, era evidente que no íbamos a tomar las imágenes tal como las tomábamos de la cámara. Tuvimos en cuenta, de que el dispositivo objetivo para desarrollar la aplicación, debía ser un móvil de gama media-baja, ya que queríamos que lo pudiese usar cualquier alumno sin precisar de un smartphone de última generación.

Comenzamos a hacer pruebas y empezamos a reducir las imágenes y llegamos a un punto dónde la aplicación en un móvil se veía genial, en una tablet se veía aceptable (contando que podríamos poner imágenes HD pero aquello encarecería el tamaño de la aplicación) y ocupaba bastante menos que el original. Finalmente, comprimimos las imágenes a un 80 por ciento de su calidad original, pasamos a una resolución de 600x400 para las fotos que cargan en solitario (actividad tres) y 400x300 para aquellas que cargaban junto a otras tres más.

El peso que llegaron a ocupar las imágenes era en algunos casos pasaba a ser unos 30-50 kilobytes frente a los dos o tres megabytes de lo que ocupaban originalmente, por lo que la ganancia era muy provechosa para el conjunto del proyecto.

Por último, los **vídeos** también necesitaron de un proceso de compresión redimensión y una codificación exacta para funcionar en Android. Por ello tuve que seguir el siguiente procedimiento por cada vídeo:

Primero, redimensionaba el vídeo a la resolución deseada. La cámara con la que grabábamos las tomas sacaba una resolución de 720x576. Como ya hemos dicho y teniendo en cuenta que se iba a utilizar en móviles, los redujimos a 320x208 para que tomaran una resolución más adecuada a la potencia de un móvil. Además, se le bajaba la tasa de bits de 256 Kbits por segundo a 151 Kbits por segundo. De igual manera que sucede con el sonido, a menos bits por segundo menos calidad pero menos iba a ocupar el vídeo final.

En ese primer proceso ya obteníamos la ganancia de peso que queríamos obtener, pero no lográbamos que los vídeos se reprodujeran en todos los dispositivos. En algunos daban fallos, en otros sólo se escuchaban... no era del todo compatible con el sistema Android. Para que un vídeo funcione en la mayoría de dispositivos Android necesita ser codificado en MP4 pero con un perfil exacto, el perfil **Baseline** y parece ser que no todos los editores de vídeo traen esa opción.

De nuevo, tuve que hacer un segundo proceso de codificación a MP4 con otro programa que no me permitía cortar trozos del vídeo pero que sí me permitía codificar en el perfil Baseline. Una vez exportado en ese perfil y en el códec y formato MP4 los vídeos se podían reproducir sin problemas en todos los dispositivos, lo que nos evitaba problemas de compatibilidad con ciertos dispositivos más anticuados.

El tamaño final de los ficheros oscila entre 100 y 200 kilobytes mientras que los ficheros originales ocupaban unos 2.5 megabytes cada uno.

Como resultado final de los tres procesos de compresión descritos con anterioridad (fotos sonidos y vídeos) pudimos bajar de unos 17 MB que ocupaba la aplicación con poca compresión y faltando algunos recursos a los 9.7MB que ocupa la última versión plenamente funcional. Mi idea inicial era que como mucho debía ocupar 20 MB la versión final, pero después de los test realizados a los alumnos vimos que teníamos que reducir aún más el peso de la aplicación por los resultados obtenidos y dejarlo cercano a los 10MB para que fuera funcional para la gran mayoría.

6.10. Puntuaciones: base de datos Sqlite3

Para guardar las puntuaciones de los alumnos, investigué maneras de guardar datos en Android. Pude haber guardado los datos que me interesasen en un fichero binario, y luego haberlo encriptado a mi antojo para almacenar los datos. Así es como se suele evitar el 'hacking' (se suele decir cuando se modifica un aspecto del juego para sacar ventaja) y impedir que un usuario tal y como empiece su juego tenga todos los apartados desbloqueados.

Dadas las circunstancias no fue necesario la encriptación de la partida guardada ya que no veíamos que alguien fuera a modificar las partidas (habría que tener amplio conocimiento informático para ello) para simplemente obtener el acceso a el chat o mejor dicho a la actividad colaborativa de manera anticipada. Por ello decidí utilizar una base de datos normal y corriente.

Una vez explicada su desventaja de ser modificable, hay que aclarar que lo más recomendado para guardar datos en una aplicación de Android es utilizar una base de datos gestionada por Sqlite3. Como no se podría considerar una 'partida guardada' que albergue muchos datos personales, simplemente le llamamos Perfil en el menú principal, ya que sus únicas dos tablas son la de usuario y puntuaciones. La aplicación no guarda por dónde te quedaste o qué actividades tienes desbloqueadas, guarda tus puntuaciones.

La tabla de usuario simplemente alberga el nombre del usuario (y ninguno más, sólo uno) y permite que cambiemos el nombre de éste a nuestro antojo desde el menú principal.

La tabla de puntuaciones alberga por cada puntuación: puntos positivos, errores cometidos, en que momento se realizó la puntuación y cuanto se tardó.

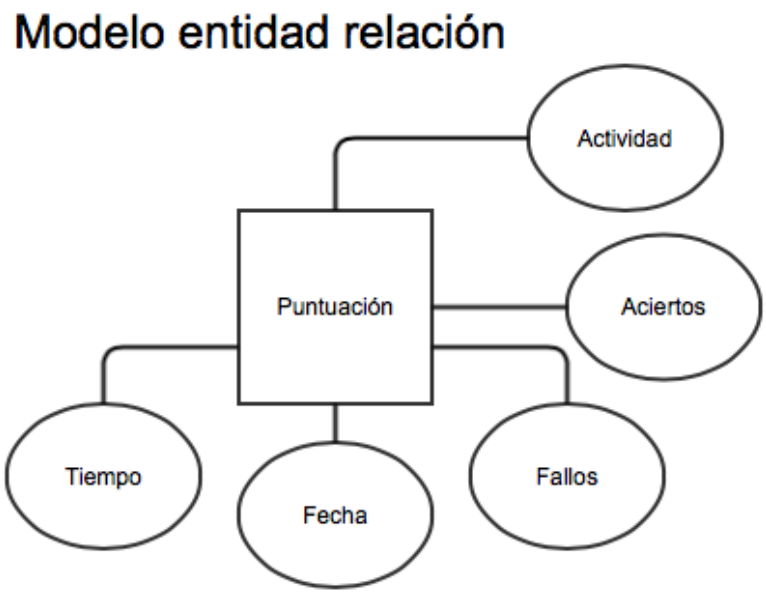


Figura 6.2: Modelo entidad relación Base de datos

Así, el menú principal comprueba las puntuaciones y sabe que nos queda aún por desbloquear para llegar al Chat, partiendo de una base de datos muy sencilla con pocos datos, que se puede observar en la (fig. 6.2).

6.11. Tareas del juego Catch me, if you can!

Para implementar las tareas de la actividad *Catch me, if you can!* pensamos en un grafo extensible por si necesitábamos implementar más tareas o modificar las actuales. Para ello y como es de costumbre en este proyecto, definimos un fichero de recursos XML que nos ayudará a construir nuestra actividad.

Vamos a analizar el fichero **assignment.xml** por partes, primero hablaremos del número de actividades que pueden surgir en la actividad colaborativa:

```
<integer name="max">2</integer>
```

En el código que aparece arriba, tenemos definido el número máximo de actividades que la actividad usará. La aplicación cuenta con dos actividades en este caso como máximo. ¿Cómo están definidas esas dos actividades? Pasamos a mostrarlo:

```
<string name="assignment1.0">
    Zeuge 1 ist im Café und trinkt Kaffee. "\n"
    Zeuge 2 ist am Strand und joggt. "\n"
</string>

<string name="assignment1.1">
    Zeuge 3 ist im Park und plaudert. "\n"
    Zeuge 4 ist in der Bibliothek und lernt. "\n"
</string>

<string name="assignment1.2">
    Zeuge 5 ist in der Kneipe und raucht. "\n"
    Zeuge 6 ist am Strand und hört Musik. "\n"
</string>

<string name="assignment1.3">
    Zeuge 7 ist im Supermarkt und kauft Kaffee. "\n"
    Zeuge 8 ist im Park und flirtet. "\n"
</string>

<string name="assignment1.4">
    Zeuge 9 ist im Café und liest Zeitung. "\n"
    Zeuge 10 ist am Strand und trinkt Bier."\n"
</string>

<string name="assignment1.5">
    Zeuge 11 ist in der Bibliothek und liest. "\n"
    Zeuge 12 ist in der Kneipe und trinkt Baileys.
</string>
```

Como se puede observar, están definidas 6 parejas de texto. Cada 'assignment1.x' define un par de indicios (que son los indicios del comisario) que dependiendo del valor de x serán de los primeros o de los últimos en aparecerle. Se puede apreciar que están numeradas y llegan desde la 1 hasta la 12 (seis pares).

¿Por qué doce y no 14? Pues sencillo, por que lo planteamos así, pero también podemos definir esa opción, ya que viene dado por el recurso 'amount':

```
<integer name="amount1">12</integer>
<integer name="amount2">12</integer>
```

También definimos el mensaje que aparecerá si un usuario logra encontrar la pista que relaciona al indicio en concreto:

```
    <string name="assignmentanswer1.1">Ja, du hast Zeuge 1 gefunden</string>
    <string name="assignmentanswer1.2">Ja, du hast Zeuge 2 gefunden</string>
    <string name="assignmentanswer1.3">Ja, du hast Zeuge 3 gefunden</string>
    <string name="assignmentanswer1.4">Ja, du hast Zeuge 4 gefunden</string>
    <string name="assignmentanswer1.5">Ja, du hast Zeuge 5 gefunden</string>
    <string name="assignmentanswer1.6">Ja, du hast Zeuge 6 gefunden</string>
    <string name="assignmentanswer1.7">Ja, du hast Zeuge 7 gefunden</string>
    <string name="assignmentanswer1.8">Ja, du hast Zeuge 8 gefunden</string>
    <string name="assignmentanswer1.9">Ja, du hast Zeuge 9 gefunden</string>
    <string name="assignmentanswer1.10">Ja, du hast Zeuge 10 gefunden</string>
    <string name="assignmentanswer1.11">Ja, du hast Zeuge 11 gefunden</string>
    <string name="assignmentanswer1.12">Ja, du hast Zeuge 12 gefunden</string>
```

Aunque nosotros lo dejamos de forma genérica, se puede modificar. Para relacionar los indicios con las pistas que van en formato vídeo, necesitamos proporcionarles un código identificador:

```
<string name="assignmentcode1.1">1001</string>
    <string name="assignmentcode1.2">1002</string>
    <string name="assignmentcode1.3">1003</string>
    <string name="assignmentcode1.4">1004</string>
    <string name="assignmentcode1.5">1005</string>
    <string name="assignmentcode1.6">1006</string>
    <string name="assignmentcode1.7">1007</string>
    <string name="assignmentcode1.8">1008</string>
    <string name="assignmentcode1.9">1009</string>
    <string name="assignmentcode1.10">1010</string>
    <string name="assignmentcode1.11">1011</string>
    <string name="assignmentcode1.12">1012</string>
    <string name="assignmentcode1.20">1020</string>
```

Y ahora mostraremos el funcionamiento de una actividad. Cuando desarrollamos la idea de que las actividades tuvieran pistas y indicios que se fueran desbloqueando, pensamos que la mejor idea sería un grafo extensible, que además nos sirviera para diseñar la secuencia del juego como quisiéramos. Aunque nuestro diseño ha sido secuencial (1,2,3...11,12) si quisiéramos reordenar las pistas y dejarlas en otro orden, sería perfectamente posible:

```
    <string name="adjacency1.1">1,2,3,4</string>
    <string name="adjacency1.2">1,2,3,4</string>
    <string name="adjacency1.3">3,4,5,6</string>
    <string name="adjacency1.4">3,4,5,6</string>
    <string name="adjacency1.5">5,6,7,8</string>
    <string name="adjacency1.6">5,6,7,8</string>
    <string name="adjacency1.7">7,8,9,10</string>
    <string name="adjacency1.8">7,8,9,10</string>
    <string name="adjacency1.9">9,10,11,12</string>
    <string name="adjacency1.10">9,10,11,12</string>
    <string name="adjacency1.11">11,12</string>
    <string name="adjacency1.12">11,12</string>
```

Se puede observar que la primera y segunda pista (adjacency1.1 y adjacency1.2) están relacionadas consigo mismas y con las dos siguientes. ¿Por qué tomamos esta decisión? Por que no veíamos justo que si uno de los dos jugadores que ejercen de policías se quedasen atascados, la actividad no pudiera continuar. Es decir, que si se encuentra una pista de una pareja, se desbloquea la siguiente. Pero esto también es modificable a gusto del programador.

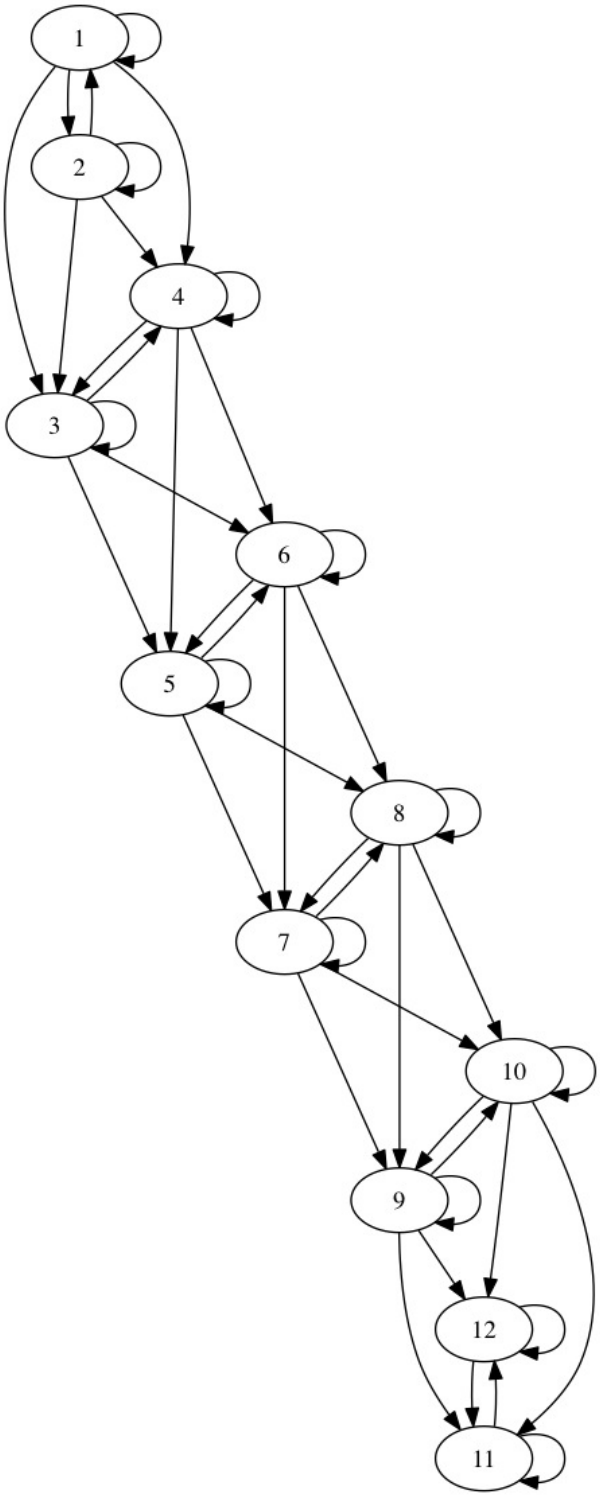


Figura 6.3: Grafo de la actividad colaborativa

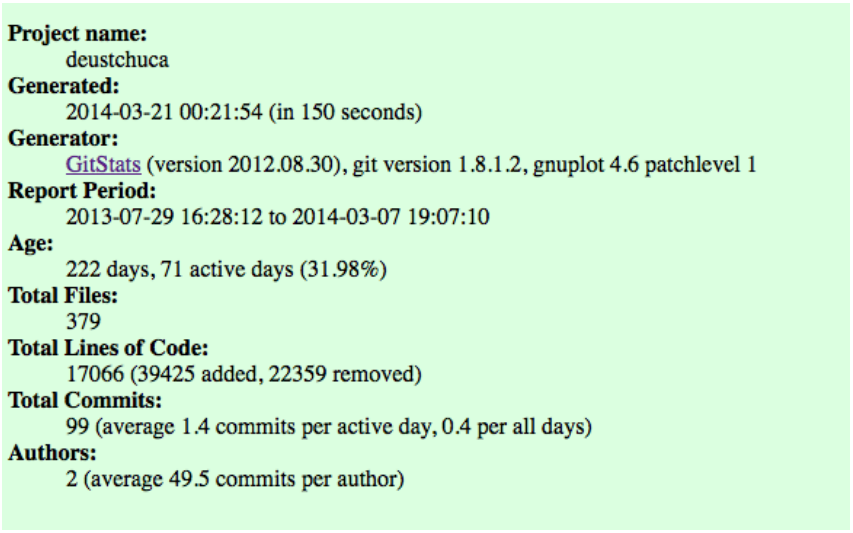


Figura 6.4: Descripción del proyecto

Si nos fijamos en la (Fig. 6.3) vemos representado el grafo gráficamente, siempre se avanza hacia adelante y no se hacen surcos hacia atrás, esto es por que nosotros decidimos que la actividad fuera siempre desbloqueando pistas nuevas y que las de mayor número fueran las que más información aportasen al jugador. La trayectoria del juego puede ser modificada a gusto del profesor en el caso de que decidiese hacer una trayectoria de juego no lineal, o si por ejemplo hubiese pistas que te llevasen a estados anteriores.

```
<string name="initialnodes1">1,2</string>
```

Por último y no menos importante, también tenemos que definir los nodos iniciales, por los cuales la actividad comienza y tomamos como punto de partida.

6.12. Estadísticas del Desarrollo

Como hemos utilizado como software de control de versiones, el sistema que propone Git, podemos sacar estadísticas del desarrollo de la aplicación mediante el programa gitstats. Con un par de comandos tenemos preparado un completo análisis sobre las estadísticas del desarrollo, que podemos ver en las siguientes capturas:

Hay detalles a tener en cuenta en la (Fig. 6.4), como que por ejemplo aparecen 2 autores, realmente es sólo un autor ya que por desgracia en mi portátil no configuré bien git y me sale con un segundo nombre, así que realmente el cien por cien de los commits realizados han sido por la misma persona. O que las líneas de código se incluyen otras librerías ya que el proyecto empezó con 4.000 líneas de código.

Hour	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
Commits	2	0	0	0	0	0	0	0	0	0	0	4	2	0	0	2	9	9	16	12	17	14	10	2
%	2.02	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4.04	2.02	0.00	0.00	2.02	9.09	9.09	16.16	12.12	17.17	14.14	10.10	2.02

Figura 6.5: Estadística sobre las horas de las aportaciones al proyecto

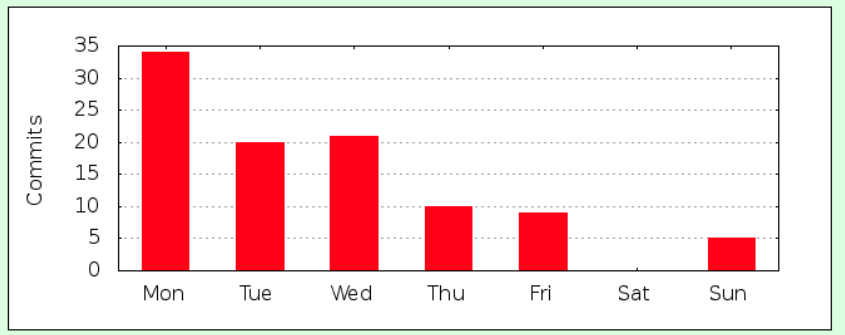


Figura 6.6: Gráfico mostrando qué días de la semana se aportaba más al proyecto

En cuanto a las horas y como se puede ver en la (Fig. 6.5), estuve trabajando de prácticas por las mañanas por lo que me fue imposible durante casi todo el desarrollo poder aportar avances al proyecto por la mañana, por eso las aportaciones son casi todas por la tarde.

En la (Fig. 6.6) podemos ver que el día que más se ha trabajado han sido los lunes, desconozco la causa de por qué habrá sido más veces los lunes que otro día de la semana, aunque si reconozco que mi dedicación al proyecto ha sido entre semana y casi nunca los fines de semana, que los utilizaba para descansar.

Durante el desarrollo del proyecto, se puede ver una bajada en el rendimiento del mismo, esto sucede como se puede apreciar en (Fig. 6.7) en el mes de diciembre. Debido a que ese mes hicimos pruebas con los alumnos, a las vacaciones y a temas personales fue imposible mantener

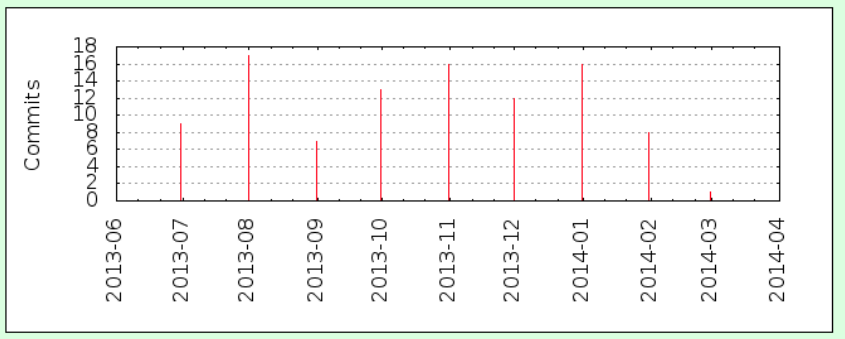


Figura 6.7: Aportaciones al proyecto según los meses

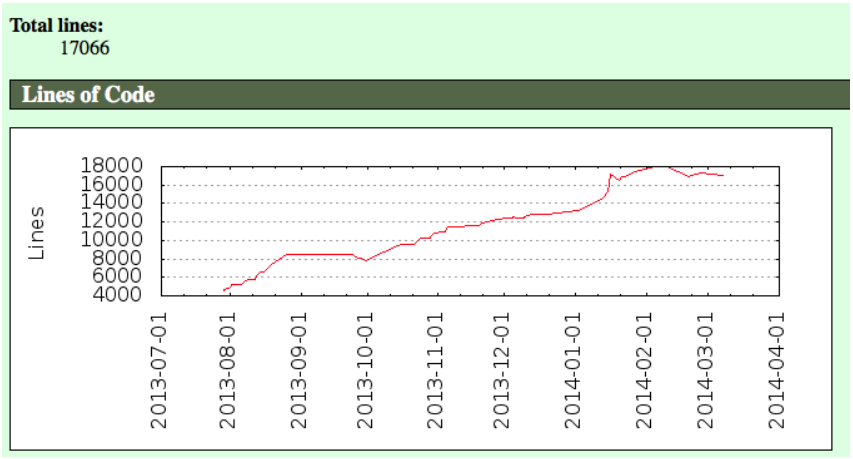


Figura 6.8: Líneas de código añadidas al proyecto

el mismo rendimiento que durante el transcurso del proyecto.

Hay un detalle más a tener en cuenta, si nos fijamos en las líneas de código, en el último trayecto del proyecto vemos que se eliminan algunas. Esto tiene una sencilla explicación, el proyecto al principio tenía también opción de chatear con una persona (esta parte se desarrolló para hacer pruebas más que para aportar funcionalidad al juego) en vez de con una sala. También tenía funciones de lista de amigos entre otras. Debido a que ya no era necesario para el programa final y para ahorrar líneas de código innecesarias éstas fueron eliminadas.

6.13. Implementar otro idioma: Recursos de texto

Ya hemos hablado de la posibilidad de adaptar a las aplicaciones Android a diferentes idiomas utilizando el directorio de recursos y usando ficheros xml. Nuestro proyecto tiene la posibilidad de ser compatible con otros idiomas mediante el uso de recursos, pero habría que tener diferentes detalles en cuenta.

Dentro del directorio 'res' tendríamos que definir un nuevo directorio dónde colocar el nuevo idioma. Vamos a tomar por ejemplo el inglés. Definiríamos un directorio llamado values-en dentro de la carpeta res en el que definiremos **todos** los recursos definidos en la carpeta values, es decir la misma estructura de ficheros. Debemos hacer la traducción de todos los recursos ya que si no podremos encontrar contenido en Alemán.

Hay que tener cuidado a la hora de traducir contenido, en los casos de las actividades *search* y *hidden* (así se llaman los ficheros xml que contienen los recursos de esas dos actividades) podríamos traducir directamente los objetos como 'Büro' por 'Office' perfectamente ya que el funcionamiento sería el mismo. Pero para las actividades tipo *fills* (así se llama el fichero de la tercera actividad) tendríamos que tener en cuenta de la construcción de los ejercicios.

El primer temario perteneciente a la actividad de rellenar los huecos tiene la forma: Ella **escucha** música en la **playa**. Si el idioma al que queremos adaptar la aplicación no tiene esa forma lingüística habría que reprogramar la actividad, modificando su vista y puede que su funcionamiento. En inglés quedaría algo así como: She **listen** music in the **beach**.

Para el segundo y tercer temario, tenemos también dos huecos a rellenar por el usuario de la siguiente forma: El delincuente tiene **un cuchillo**. Los dos huecos a rellenar por el usuario serían 'un' y 'cuchillo', por lo que en inglés tendríamos que traducirlo para que quede de la siguiente manera: The offender has **a knife**. Tendremos que fijarnos en el fichero de recursos y traducir cada parte de la frase.

Una vez traducidas las actividades, se podría traducir el modo *Catch me if you can*, para ello habrá que modificar texto y vídeos.
Por parte de los textos tendríamos que fijarnos en el fichero assignment ya que es el que contiene los indicios que recibirá el comisario. Cuando el comisario juega con sus compañeros, a medida que avancen en el juego, recibirá más indicios por parte de la aplicación. Estos indicios son del estilo 'El testigo está leyendo en el parque'. Para que aparezca en el idioma que queremos, debemos traducir uno por uno los indicios, en este caso tendríamos que poner en el recurso citado 'The witness is reading in the Park' y nos aparecería ese indicio en inglés a la hora de jugar. Si no alteramos los murales de los códigos, no hará falta modificar más texto con respecto al chat.

El resto de ficheros deben traducirse aunque no precisan de una atención especial ya que no va a variar el funcionamiento de la aplicación por el cambio de idioma.

6.14. Implementar otro idioma: Recursos de sonido y vídeo

Los recursos de sonido son otro aspecto muy importante del temario de la aplicación. Algunos sonidos contendrán preguntas que se le realizan al alumno y otros contendrán respuestas. Los sonidos creados para la aplicación se encuentran en el directorio 'raw' dentro de res. Android no da solución oficial a como utilizar diferentes recursos de sonido y de vídeo para diferentes idiomas, pero nosotros hemos implementado una alternativa para que si se puedan adaptar los sonidos y los vídeos muy sencilla.

Cuando se traduzcan los ficheros de recursos, en el fichero strings se puede observar que hay un recurso llamado 'prefix', como su nombre indica, es un prefijo para detectar los recursos de audio y de vídeo que queramos que se reproduzcan en nuestro idioma. Es decir, en el recurso 'prefix' podemos poner como prefijo (recomendamos) 'en_' para definir que nuestros recursos de audio y de vídeo empezarán por en_ y tendrán el mismo nombre que los ficheros originales. Es decir, que si hay un fichero de audio llamado fills_faces.mp3 en el idioma por defecto, nosotros crearemos uno llamado en_fills_faces.mp3 y definiremos la cadena 'prefix' como 'en_' para que el sistema sepa cual de los sonidos cargar.

Para los recursos de vídeo tenemos el mismo problema que con los de audio, y la solución pasa por ser la misma exactamente, modificar el recurso 'prefix' y añadir recursos de vídeo con el prefijo que simbolice el idioma que queramos adaptar. Hay que recordar que los ficheros de vídeo son las pistas que se obtienen en el apartado *Catch me if you can* y que son el medio por el cual los jugadores resuelven quién es el *asesino ficticio*, por lo que es imprescindible que sea traducido correctamente.

Capítulo 7

Pruebas

7.1. Pruebas del sistema

En este apartado se detallarán las pruebas realizadas al sistema y los resultados obtenidos de éstas. Como hemos tenido una planificación por iteraciones tenemos cuatro subapartados de pruebas que corresponden a las iteraciones propias que ha tenido el sistema y por último las pruebas de aceptación que contienen las pruebas realizadas con los alumnos.

Las pruebas que hemos realizado se han realizado de manera manual y sin ningún script de automatización de pruebas ya que una aplicación de chat que se ejecute dentro del sistema Android necesitará de una serie de pruebas de interacción con la aplicación que son más sencillas de realizar directamente.

7.2. Pruebas I

En la primera instancia de la aplicación, sólo teníamos desarrollado un chat muy básico mediante el cual podíamos hacer conexión, mandar y recibir mensajes de texto. Se le realizó diversas pruebas de unitarias ya que aún no estaba desarrollado la actividad *Catch me if you can*.

Las pruebas que le realizamos fueron diversas, por ejemplo probamos el envío y la recepción de mensajes, para ello usamos un emulador android, una tablet y un dispositivo móvil. A ésta iteración se le sometió a pruebas de giros de pantalla del propio dispositivo y a pasar de primer plano a segundo plano la aplicación. También a desconectar y volver a conectar el dispositivo a la red para comprobar su funcionamiento.

7.3. Pruebas II

En esta iteración, se probaba por un lado la integración y funcionalidad del sistema de chat y por otro lado las pruebas unitarias de las actividades. Para ello, probábamos los códigos desde la conversación chat. Probábamos a almacenar pistas y a guardarlas y a participar en partidas de

prueba en las que se pudiera probar la integración de las distintas partes de las que se compone el chat. En definitiva se probó el conjunto de todas las funcionalidades del chat puestas entre sí.

Por el lado de las actividades, fuimos probando la funcionalidad de los botones de juego y la correcta visualización de las imágenes. Así como la reproducción de sonido para servir de ayuda al usuario.

7.4. Pruebas III

Siendo la tercera iteración ya teníamos que realizar pruebas no funcionales del apartado chat (que ya estaba bastante refinado) para ver si funcionaba correctamente en los dispositivos de los alumnos. Gracias a estas pruebas nos dimos cuenta de que una mala cobertura Wifi podría echar por tierra todo el trabajo con la aplicación, y recomendamos su uso con conexión 3G que aunque lenta, puede ser más estable que una conexión Wifi lejana.

También hicimos pruebas escaneando códigos con las cámaras de sus dispositivos y pudimos notar que bajo ciertas condiciones lumínicas se dificulta la obtención del código del papel, aunque esto se soluciona en un par de intentos y no destroza la partida.

Por la parte de las actividades hicimos pruebas de integración realizando partidas completas para poder comprobar su correcto funcionamiento con todas las funcionalidades. Gracias a estas pruebas obtuvimos información importante sobre la curva de dificultad que queríamos proponer al alumno y sobre la dificultad del manejo de la propia actividad.

7.5. Pruebas IV

Una vez finalizado el desarrollo del apartado chat, se realizaron pruebas de aceptación que detallaremos más en el apartado siguiente. El objetivo de esta prueba era comprobar el uso de la aplicación por parte de usuarios reales y se utilizaron los dispositivos de los propios alumnos en una prueba real sobre el sistema.

Las pruebas funcionales que le realizamos a los menús del sistema y las actividades fueron ver que cumplían con sus casos de uso propuestos.

Las pruebas de no funcionales de las actividades las pudimos realizar con la instalación de la aplicación en diferentes dispositivos y obtuvimos una respuesta positiva del comportamiento del sistema en dispositivos de gama baja o más obsoletos, aunque tuvimos problemas con uno de ellos que era el de menor rendimiento.

7.6. Pruebas de Aceptación: Catch me if you can

Las pruebas de aceptación que realizamos se realizaron con diferentes alumnos en la facultad de Filosofía y Letras de Cádiz en dos tardes diferentes.

En la primera tarde, tuvimos grandes problemas con la conectividad de la aplicación debido a la red pública Wifi de la UCA, ya que ésta no nos permite estar mucho tiempo conectados sin volver a acreditarnos. Por si fuera poco, el alcance de la red Wifi en algunas zonas del edificio era escasa o nula, por lo que en el momento en el que un usuario perdía la conexión, peligraba el desarrollo de la prueba. Cuando el jugador que ejercía de comisario se desconectaba de la red, había que comenzar completamente de nuevo.

Aún así casi logran resolver todos los requisitos de la aplicación (resolvieron 10 de 12 pruebas) y ya con la partida terminada por desconexión el jugador pudo resolver quién de los sospechosos era el correcto. Debido a esta clase de infortunios tuvimos que recomendar el uso de una red móvil 3G para los dispositivos que fueran a utilizar la aplicación, y a mudar el servidor a uno público para que se pudiera acceder desde cualquier parte.

En la segunda tarde, después de haber solventado ciertos errores de conexión que nos afectaban en la primera tarde, pudimos probar la aceptación del sistema por parte de los alumnos. Los resultados recogidos fueron positivos, de dos partidas realizadas con tres alumnos, en los que cambiamos los roles, obtuvimos dos victorias con la obtención de una serie de pruebas. El manejo de la conversación chat resultaba intuitivo para los jugadores y supieron adaptarse rápidamente al sistema de juego, aunque éste requería una gran atención por parte del jugador (hay que tener en cuenta que ese era nuestro propósito).

7.7. Pruebas de Aceptación: Actividades y uso general de la App

Las pruebas de aceptación para las actividades se realizaron por parte de un gran número de jugadores. Desde familiares que ofrecieron jugar a la aplicación, siguiendo por la directora Anke (que resolvió la aplicación entera numerosas veces), siguiendo por mi y acabando por un grupo de 12 alumnos del instituto IES Poeta García Gutiérrez pertenecientes a educación secundaria.

En las pruebas realizadas en colaboración con el instituto, pudimos probar el uso de la aplicación durante hora y media de juego con los alumnos. De los doce propuestos, dos superaron **todas** las actividades sin ayuda ninguna y sin necesitar un repaso de sus conocimientos. Nueve de ellos fueron capaces de jugar hasta la última o penúltima actividad del juego sin grandes complicaciones aunque no fueron capaces de superar la última. Y sólo un jugador no mostró gran interés en superarse pasada la tercera actividad del juego.

Las opiniones recogidas sobre las actividades de la aplicación fueron muy positivas y los alum-

Figura 7.1: Fotografía tomada durante la prueba con los alumnos en la facultad de Filosofía y Letras

nos mostraron su interés en participar en más iniciativas como las que proponíamos para servir de ayuda al aprendizaje del idioma.

7.8. Pruebas de validación

Una vez terminado por completo el desarrollo de la App, se distribuyó la aplicación a un grupo que superaba los ochenta estudiantes. De este grupo, sólo unos pocos (menos de diez) pudieron jugar por que poseían un sistema diferente a Android. Otra cifra parecida presentaban problemas de descarga/instalación de la aplicación, por lo que la instalación satisfactoria de la app fue realizada sobre un grupo alrededor de los sesenta alumnos.

De tal grupo, pudieron probar la aplicación (algunos con cascos para poder escuchar mejor lo que le decía el programa) durante unos cuarenta minutos, por lo que fueron capaces de llegar a la tercera actividad sin grandes problemas. Cabe destacar que sólo hubo que explicar a dos jugadores que debían superar una puntuación al menos tres veces (o dos si es actividad múltiplo de tres) para poder avanzar en la aplicación. También se valoró muy positivamente la inclusión del apartado Punktezahl, ya que en éste los alumnos podían consultar sus avances y comprobar si les quedaba algún intento por aprobar.

Figura 7.2: Fotografía tomada durante la prueba con los alumnos en la facultad de Filosofía y Letras

La acogida general fue bastante positiva y mostraron gran interés en querer seguir utilizando la aplicación fuera de la clase como método de repaso (y de superarse a sí mismos como si fuese cualquier otro videojuego) para el aprendizaje de la lengua.

Capítulo 8

Conclusiones

8.1. Trabajos futuros

VocabTrainer A1 saca partido a ciertas características del móvil, como puede ser la cámara para reconocer códigos en objetos, o el altavoz para reproducir la correcta pronunciación de una palabra en la lengua extranjera. Pero hay cosas que podrían complementar perfectamente a la aplicación y que no hemos desarrollado por que llevaría mucho tiempo poder implementarlas.

8.2. Analizar la pronunciación del jugador

De manera parecida a lo que hace Google, podríamos haber enfocado el aprendizaje también a la pronunciación. Cuando usas un móvil con Android relativamente moderno, existe la opción de dictarle palabras por voz. Aunque no funciona completamente bien en todos los casos, es capaz de escribir por ti lo que estás diciendo verbalmente.

Podría ser interesante, que se le pidiera al alumno que pronunciara ciertas palabras y que la app pudiese compararlas con una base de datos online (o utilizar algún servicio de google si lo hubiera) para poder detectar qué palabra ha pronunciado. Así podríamos valorar la pronunciación del alumno, aunque estas herramientas no están del todo pulidas y si es un extranjero el que pronuncia un idioma puede dar más problemas aún.

8.3. Reconocimiento de objetos

Aunque no soy ningún experto en el tema, he visto compañeros implementar reconocimiento de imágenes para reconocer matrículas y formas básicas. Dado que nosotros simplificamos este problema con el tema de los códigos, no seguimos profundizando en el reconocimiento de imágenes que sirvieran como actividad. Pero eso no quiere decir que no tuviese cabida un reto que lo implementara.

Con la debida experiencia, se le puede pedir al alumno, al igual que ha sabido reconocer ciertos lugares en imágenes, que haga fotos de ciertos objetos básicos. No hablo de pedirle que fotografíe una playa y que el móvil lo analice por que eso tendría mucha dificultad, pero si podríamos pedirle que fotografiara un círculo, un triángulo, un cuadrado... el número 34 (por decir un número), la palabra *alemán*. Podría ser vocabulario básico que encajase perfectamente en el A1 ademas de ser muy divertido tener como tarea escanear ciertos objetos, palabras o números que nos encontremos por la calle o en casa.

Este método sacaría partido de la gamificación, ya que mediante el juego tan sencillo de hacerle una foto a un objeto de la calle nos serviría para aprender el vocabulario de lo que nos rodea en la lengua extranjera.

8.4. Cumpliendo los objetivos

Cuando empezamos el proyecto teníamos como objetivo dotar de una herramienta que sacara partido de la tecnología de los teléfonos móviles. Para ello pensamos en diversas vías como podía ser la realidad aumentada, los mundos virtuales... pero debido a su dificultad y a su complicado mantenimiento fuimos derivando la idea al reconocimiento de códigos para reconocer objetos que estaban en el vocabulario a aprender por los alumnos.

Al principio la idea era muy ambiciosa técnicamente, pero derivamos esa dificultad a una sencillez de uso, queríamos que fuera fácil y sobre todo muy intuitiva de usar. Queríamos que el alumno supiera superar el reto que se le proponía pero sin tener que pararse a pensar en cómo jugarlo, si no centrarse en el idioma plenamente. La dificultad tenía que residir en el contenido y no en el modo de juego. Es posible que la actividad chat plantee alguna duda de uso que puede ser resuelta teniendo a alguien al lado que ya haya utilizado la aplicación, pero por el resto la aplicación es bastante intuitiva y creo que en ese detalle hemos cumplido de sobra, los alumnos que la han probado no han presentado problemas a la hora de jugarla.

Como método de aprendizaje, el tiempo dirá si realmente ayudó o no ayudó a aprender un idioma, la opinión de los alumnos es que sirve para repasar contenido allá dónde estés y que incita a querer superarse uno mismo, quizá el producto final que hemos sacado sea un poco diferente a como lo planteamos en un principio, pero hemos cumplido los objetivos de sacarle partido a la movilidad y a la gamificación.

Repasando un poco los objetivos del apartado 1.2 , podemos decir que si que hemos proporcionado una fuente fiable de conocimientos de Alemán, ya que son conocimientos contrastados, de eso no hay duda. La capacidad de autoevaluación creo que es fácilmente demostrable, la misma aplicación te dice si has sido capaz de superar un temario sencillo o en caso contrario te dice dónde te has equivocado sin depender de un tercero.

Los otros tres puntos se escapan un poco a mi labor como diseñador y programador de la aplicación, creo que como ya he dicho el tiempo será el que dirá si se ha conseguido que la evaluación

docente y el trabajo en equipo se desarrolla mejor con actividades como la que propone nuestra aplicación.

8.5. Análisis de la experiencia

A la hora de desarrollar la aplicación en solitario (en cuanto al plano técnico) me he encontrado con numerosas dificultades y impresiones por el camino. Algunas de ellas me las esperaba y otras no me las esperaba tanto. Algunas han sido positivas y otras no tan positivas. Mejor comentarlas al detalle.

Aspectos positivos

Al desarrollar una tecnología que está al día (Android) me encontré con una cantidad increíble de documentación que me facilitó el aprendizaje, que fue muy rápido a la hora de programar en la plataforma.

Programar algo que puedes tocar y escuchar llega a ser muy divertido y siempre puedes enseñarlo a la gente que te rodea. Estás creando algo que a la gente le interesa y lo puede probar fácilmente, en cambio si hubiese sido otro tipo de proyecto quedaría más en el terreno profesional.

Tener a mano tal cantidad de herramientas (Bases de datos, manejo de vídeos, manejo de imágenes, manejo de sonido, manejo de recursos... etc) hace que todo esté más estandarizado, y que la implementación de contenido sea mucho más directa e intuitiva. Nada de tener que utilizar una librería gráfica en concreto, de tener que descargarse una librería que implemente una base de datos... todo lo tienes empaquetado en el mismo lugar y garantizado su funcionamiento. En ese sentido el desarrollo de la app ha sido más liviano de lo que podía haber sido.

Aspectos negativos

Me ha llevado mucho tiempo algunos aspectos que hoy día se dan por básicos. Hoy día parece sencillo el desarrollo de un chat. En cambio me sorprende que haya tan poca documentación y herramientas para desarrollar uno en condiciones. Y aún así tengo que agradecer que la librería me facilitó algo las cosas, pero me llevó mucho tiempo desarrollar un chat estable y que fuera completamente funcional.

El problema de la conectividad, tuvimos numerosos problemas a la hora de conectar la aplicación con el servidor. Primero usamos un servidor público para hacer pruebas de conexión (que luego descubrí que el servidor se caía cada dos por tres) luego mi ordenador de casa pero sólo para la conexión Wifi (por lo que por 3g no pude hacer pruebas) más tarde instalamos un servidor en la red de la UCA, que eso supuso otro problema.

La red Wifi de la UCA te expulsa cada x minutos para que vuelvas a introducir los datos de usuario para poder seguir conectado, y no íbamos a utilizar la red privada dado que supone mucha pérdida de tiempo instalarla en varios dispositivos a la vez. Por lo que tuvimos que utilizar un servidor para nuestro proyecto que estuviera alojado fuera de la UCA y se pudiera dar uso mediante 3G, ya que aunque instaláramos la red privada de la UCA había zonas sin cobertura Wifi que hacían que la partida se terminara de repente. En ese sentido fue un absoluto caos.

Como experiencia

Como experiencia he tenido también aspectos positivos y otros negativos. Los positivos me han hecho crecer como persona, y los negativos me han costado los ánimos más de una vez. Me gustaría detallarlos un poco más:

Experiencia positiva

Antes de empezar a desarrollar para Android, no había hecho ningún proyecto de una tal tamaño. El simple hecho de pensar en un proyecto final de carrera me asustaba un poco y pensaba que podría ser muy difícil. En algunos puntos lo ha sido, pero siempre he tirado hacia adelante e intentado arreglar todo lo posible. A día de hoy no le tengo el miedo que tenía antes a los proyectos. Creo que he ganado mucha confianza en lo que puedo dar de mi mismo, y eso es algo que no tiene precio.

Aunque la idea de hacer la aplicación era para hacer mi PFC (y mi motivación junto al mundo de las tecnologías) nunca antes había programado para un gran número de personas. Me imaginaba que la aplicación la acabarían usando los alumnos en clase de manera obligatoria, pero no me imaginaba tal aceptación positiva. Haber programado para las personas y que éstas te den las gracias por hacer algo divertido para ellas me ha servido como recompensa, no creía que fuese a suceder de esa manera, ni que me fuese a sentir tan bien con ello.

Tener la responsabilidad de un desarrollo a tus espaldas te enseña a ser disciplinario, a corregir los errores, a mejorar lo que veas mal, a pensar en tus ratos libres que podrías añadirle. En cierto modo me ha hecho demostrarme a mi mismo que soy capaz de trabajar con otros y cumplir con mi parte del trabajo tal y como los demás hacen. He aprendido a trabajar coordinadamente y marcarme unos objetivos en grupo y cumplirlos.

Experiencia no tan positiva

Aunque no puedo considerar que haya sido algo negativo, si ha habido aspectos duros en al experiencia.

Ha habido veces que el trabajo ha desbordado e invadido parte de mi tiempo libre, que se ha visto mermado casi hasta el punto de ser nulo algunas semanas. Aunque hemos llevado un ritmo de trabajo muy bueno y hemos obtenido muy buenos resultados, ha habido ocasiones en las

que hemos tenido que sacrificar mucho tiempo para conseguir avances. No digo que no haya merecido la pena, pero no ha sido agradable en ciertas ocasiones y creo que de eso he aprendido para que no se repita muchas veces.

Algunas veces me he sentido impotente ante un fallo al que no encuentras solución durante horas e incluso días, y por mala suerte el fallo que intentas solucionar parece ser que sólo lo tienes tu (o al menos nadie se ha decidido a comentarlo en la red) y tienes que investigar tus propias soluciones y luchar contra los ánimos de dejemos esto por hoy y descansemos un rato.

Al igual que ha habido muchísima gente que ha sabido reconocer el trabajo que el proyecto lleva detrás, no todo el mundo es capaz de apreciar el esfuerzo que tiene investigar una característica que hoy día tiene toda App de móvil, lo que para muchos puede ser intuitivo (como enviar un mensaje de texto) puede llevar meses de trabajo detrás para otros.

Parte III

Apéndice

Apéndice A

Manual de instalación

A.1. Instalación y manejo Openfire

Para el correcto funcionamiento de la aplicación necesitamos un servidor Openfire funcionando para darnos servicios de mensajería a la aplicación. Sin este software no seríamos capaces de realizar la actividad *Catch me if you can*, ni de enviar mensajes ni siquiera de conectarnos.

Pasamos a describir los pasos necesarios.

A.1.1. Requisitos Openfire

Como todo software, necesitaremos correrlo sobre una máquina con unos requisitos necesarios para el funcionamiento del mismo. Dependiendo de nuestras exigencias serán más altos o más bajos los requisitos.

La instalación de Openfire tiene unos requisitos sacados de la página oficial que son los siguientes:

- Tener instalado Java 5, aunque es recomendado tener instalado **Java 6**.
- Base de datos, preferiblemente **MySQL**.

También será necesario tener instalado el software de Apache2, PHP y PHPMyAdmin para poder gestionarlo vía web.

En cuanto a el sistema operativo sobre el que va a funcionar necesitamos una de las siguientes opciones:

- Windows 2000, Windows Xp, Windows Vista, Windows Server 2003, Windows server 2008
- Linux, Solaris.

- Mac OS X

Y ahora un punto importante es la potencia que tendrá el servidor, dependiendo de la cantidad de usuarios necesitaremos una potencia dada:

- **Hasta 500 usuarios** Al menos 384 MB de Ram disponible y un procesador de 1.5 GHz.
- **Hasta 10.000 usuarios** Al menos 768 MB de Ram disponible y un procesador de 3.0 GHz.

El resto de requisitos son para más de 25.000 usuarios, por lo que vamos a tomar como referencia los datos mencionados arriba como requisitos recomendados para la instalación de un servidor para una base de usuarios no muy grande, ya que nuestro proyecto no tiene un propósito mayor.

A.1.2. Instalación Openfire

Pasamos a describir los pasos de la instalación de Openfire en un sistema operativo Ubuntu (sistema operativo utilizado para desarrollar el proyecto, para el servidor de pruebas, y para el servidor que tenemos actualmente).

Primero instalaremos el software necesario para hacer funcionar openfire, Apache2 + MySQL + PHP5 Y PHPMyAdmin. Tendremos que introducir en la terminal de Linux las siguientes ordenes.

- sudo apt-get -y install apache2
- sudo apt-get -y install mysql-server mysql-common
- sudo apt-get -y install php5 php5-cli
- sudo apt-get -y install phpmyadmin

Una vez instalados los paquetes necesarios, debemos de configurar ciertos aspectos.

Para que apache2 muestre el error de host:

- sudo echo “ServerName localhost” >> /etc/apache2/httpd.conf

Para que Apache2 tenga los caracteres propios de nuestra lengua:

- sudo echo “AddDefaultCharset ISO-8859-1” >> /etc/apache2/conf.d/charset

Por último debemos resetear Apache:

- sudo /etc/init.d/apache2 restart

Ahora procederemos a la instalación de Java 6, si es que no lo tenemos aún:

- sudo apt-get install sun-java6-bin

Lo configuramos como interprete principal de java:

- sudo update-alternatives –config java

Ahora añadimos el usuario necesario para openfire:

- sudo adduser openfire

Descargamos el paquete de instalación (sustituir por última versión):

- wget -c http://www.igniterealtime.org/downloads/download-landing.jsp?file=openfire/openfire_3.7.0_al

Una vez descargado, instalamos el paquete:

- sudo dpkg -i openfire_3.7.0_all.deb

Ahora copiamos el contenido básico para Openfire y MySQL:

- sudo cp /usr/share/openfire/resources/database/openfire_mysql.sql $HOME/
- sudo chmod 777 openfire_mysql.sql

Uno de los pasos más importantes es crear una base de datos para que openfire funcione correctamente:

- mysqladmin -h localhost -u root -p create openfire
- mysql -h localhost -u root -p openfire <openfire_mysql.sql

Una vez creada la base de datos ahora podemos crear un usuario y asignar permisos en MySQL:

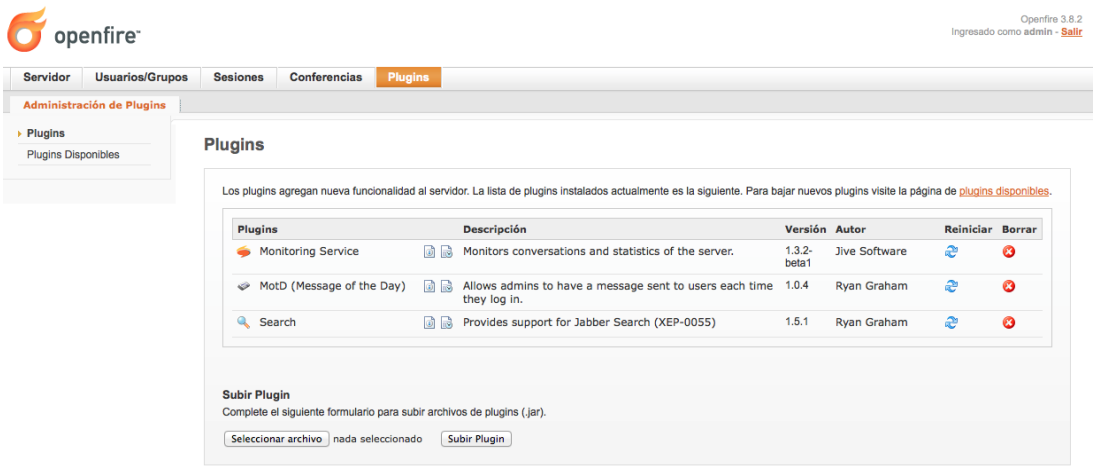


Figura A.1: Pestaña plugin del servidor Openfire

– Linea=”CREATE USER openfire@localhost IDENTIFIED BY ’CONTRASEÑA’;” echo “$Linea” | mysql -h localhost -u root -p

– Linea="GRANT ALL ON openfire.* TO openfire@localhost;.echo “$Linea"| mysql -h localhost -u root -p

Ahora eliminamos los archivos que sobran de la instalación:

– rm openfire_3.7.0_all.deb

– rm openfire_mysql.sql

Reseteamos Openfire para empezar a funcionar con la nueva configuración:

– /etc/init.d/openfire restart

Una vez concluidos estos pasos, ya debemos de tener funcionado nuestro servidor openfire en la máquina. Para administrar el servidor vía web debemos abrir un navegador y color la siguiente url: *http://ip:9090*

La ip será la dirección del servidor en la que hayamos instalado el software Openfire y el puerto 9090 es el puerto por el cual se administra el servidor via web.

A.1.3. Guardar conversaciones

Uno de los aspectos más importantes del proyecto es la capacidad que tiene para poder almacenar la información de las conversaciones realizadas entre los alumnos. Para ello el servidor

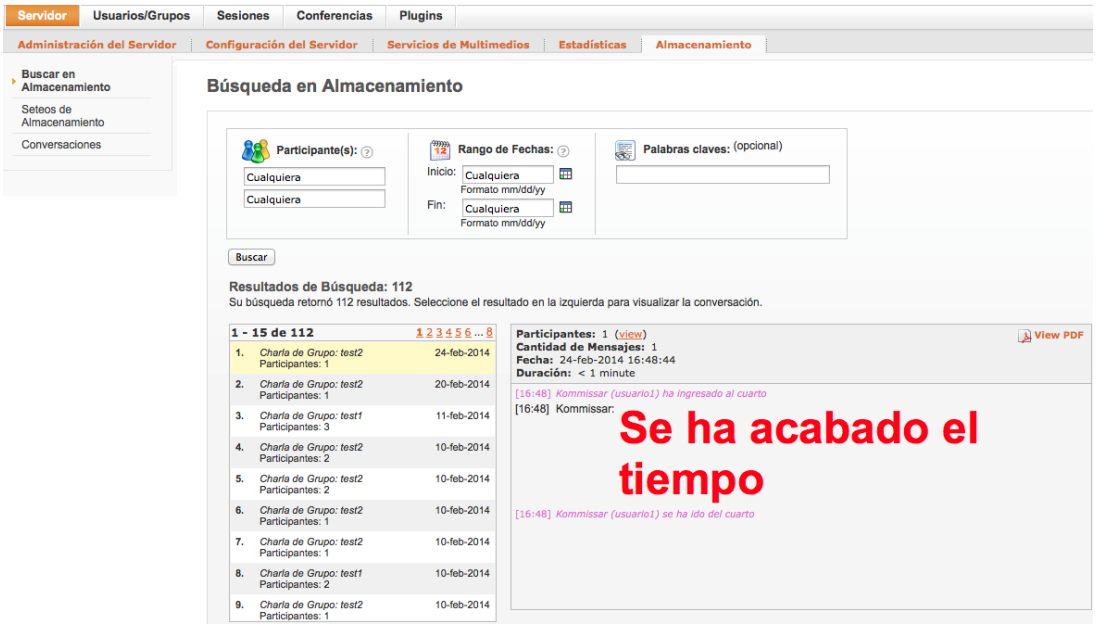


Figura A.2: Pestaña de almacenamiento del servidor Openfire

necesita de un plugin que gestione el almacenamiento de las conversaciones ya que por si mismo no tiene la capacidad para mostrar y gestionarlo.

Para configurar el almacenamiento de mensajes primero tenemos que instalar el plugin tal y como vemos en la (Fig. A.1) :

- Accedemos a la pestaña plugin del servidor Openfire.
- Seleccionamos la opción plugins disponibles.
- Buscamos en la lista el plugin llamado Monitoring Service.
- Pulsamos sobre el aspa verde debajo de la columna ‘Instalar’.

Una vez instalado el plugin, nos vamos a la sección **Servidor** y buscamos la última opción llamada Almacenamiento (Fig. A.2) .

Debemos entrar en el apartado **Seteos de almacenamiento** y dentro de esa vista configurar el plugin a nuestra conveniencia. Lo ideal sería activar las charlas de grupos ya que precisamente eso es lo que nos interesa para el proyecto, pe
ro podremos activar todo tipo conversaciones aunque éstas no vayan a servirnos para éste propósito.
Como se puede ver en la (Fig. A.2) podemos consultar por usuario, por fecha o por palabras que se hayan utilizado en la conversación. El plugin nos mostrará una serie de conversaciones que coinciden con nuestros parámetros de búsqueda. Podemos seleccionar la deseada y visualizarla

desde el recuadro de la derecha.

La opción más interesante viene ahora ya que una vez seleccionada una conversación podemos pulsar el botón View PDF y nos descargará un pdf con la conversación de manera que podamos archivarlo dónde más nos convenga e imprimirlo para poder dárselo al alumno.

Apéndice B

Manual de usuario

En este apartado se va a explicar el uso de la aplicación enfocado al usuario, así como su instalación y configuración propia.

B.1. Requisitos

Antes de instalar debemos de cumplir unos requisitos mínimos para la instalación de Vocab-Trainer A1 en nuestro dispositivo Android. Para poder ejecutarse la aplicación necesitará que el sistema operativo Android sea mínimo la versión 2.3, ya que ésta ha sido elegida como la versión mínima para el desarrollo del mismo. Aparte, serán necesarios 10MB de espacio libre para la instalación de la aplicación (Barcode Scanner va aparte).

Para el correcto funcionamiento de la lectura de códigos, necesitaremos una cámara integrada en el dispositivo y tener espacio disponible para Barcode Scanner. La aplicación podrá funcionar sin éste requisito pero no se podrá llevar a cabo el uso de la actividad *Catch me if you can*.

Como resumen, se requerirá:

- Al menos **10MB** de espacio libre en el dispositivo.
- Android 2.3 como mínimo.
- Cámara integrada en el dispositivo.
- La aplicación consumirá unos 25MB de memoria RAM.

B.2. Instalación

Como podemos observar en la (fig. B.1) una vez descargado el fichero de la instalación que tendrá extensión apk, se nos dará la opción de instalar la aplicación. Si al pulsar instalar el proceso de instalación nos avisa de que no se puede instalar la aplicación debido a que proviene de un

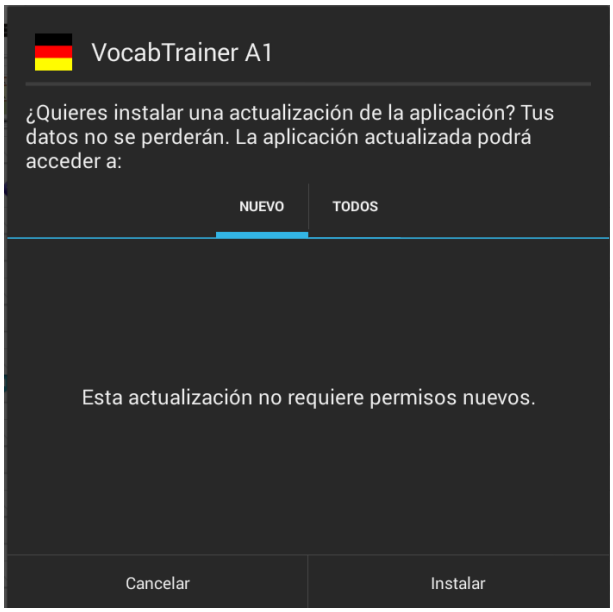


Figura B.1: Instalación de VocabTrainer A1

origen desconocido, deberemos irnos al apartado **Seguridad** dentro de los ajustes de nuestro sistema Android.

En la (fig B.2) podemos observar las distintas opciones del apartado Seguridad, debemos activar la casilla **Orígenes desconocidos** para permitir la instalación de la aplicación. Después de su instalación no será necesario tener activada esta casilla, y recomendamos que se vuelva a desactivar para evitar la instalación de software malintencionado.

Una vez activado, volvemos al proceso de instalación abriendo de nuevo el fichero APK para pulsar de nuevo sobre instalar, y se procederá a la instalación del sistema. Si todo a ido bien podremos pulsar *Listo* ó *Abrir* para empezar a probarla. Si elegimos Abrir se mostrará el menú principal (fig. B.3).

Para el completo funcionamiento de la actividad *Catch me if you can* necesitaremos tener instalado la aplicación Barcode Scanner. Para ello hemos facilitado la instalación de ésta mediante una herramienta que automatiza en gran parte el proceso. Pulsaremos el botón 'Scan' del menú principal y se nos abrirá un diálogo de pregunta sobre si deseamos instalar la aplicación o no. Pulsamos sobre el botón *yes* y se nos enviará a la aplicación Google play.

Como requisito para descargarse la aplicación, es necesario estar dado de alta en la red de Google play, así que de ahora en adelante daremos por hecho de que tenemos una cuenta en Google play para poder descargar aplicaciones. Para instalar la aplicación Barcode Scanner, sólo hay que pulsar el botón *Instalar*, por cierto, la aplicación es totalmente gratuita y no se hará ningún cargo.

Figura B.2: Apartado Seguridad del sistema Android

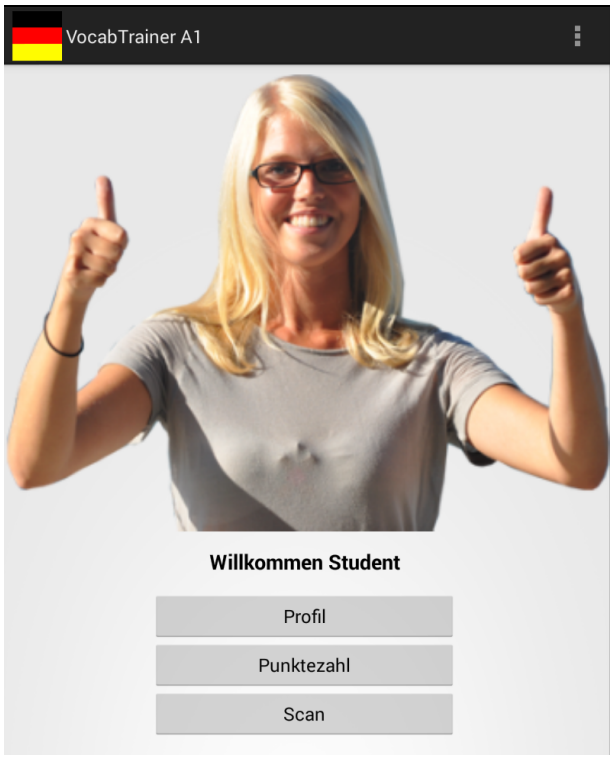


Figura B.3: Menú principal del sistema

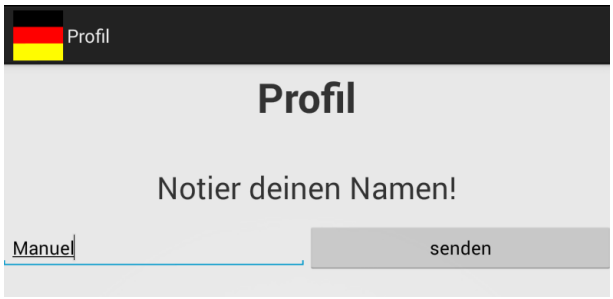


Figura B.4: Menú personalización Perfil

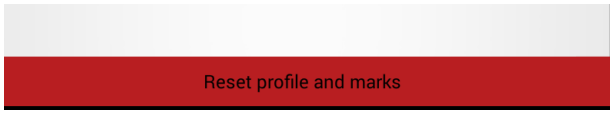


Figura B.5: Menú personalización Perfil

Una vez instalada la aplicación para poder escanear códigos Qr, debemos volver a la aplicación VocabTrainer A1 y poder poner en funcionamiento el sistema.

B.3. Configuración del perfil

Si ya os habéis fijado, os daréis cuenta de que en el menú principal se le recibe al jugador con el nombre *student*, si deseamos cambiar el nombre, podremos hacerlo pulsando el botón Profil. Una vez pulsado, dentro nos encontraremos el recuadro dónde poder escribir nuestro nombre (en el caso de ejemplo ponemos Manuel) y si pulsamos el botón *senden* se producirá el cambio de nombre del perfil.

Aparte, ésta vista contiene la posibilidad de borrar nuestros avances en el juego por si fuera necesario o por si quisiéramos repetir la experiencia desde el principio. Para ello basta con pulsar el botón rojo *Reset profile and marks* que aparece en la (fig. B.5) y contestar afirmativamente a la pregunta de si deseamos borrar todas las partidas del juego ademas del nombre de perfil.

B.4. Uso de las Actividades

Para desbloquear todas las actividades que tiene el sistema, debemos superar un número de veces cada actividad. Si las actividades están terminadas en .1 ó .2, habrá que superarlas tres veces cada una con un balance de 7 puntos positivos. Si la actividad está terminada en .3 entonces sólo

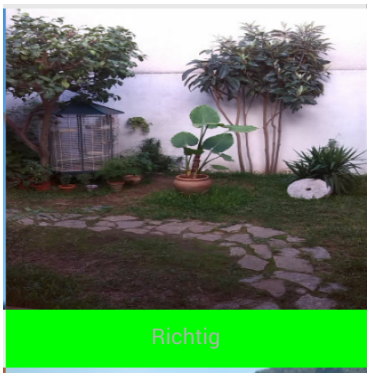


Figura B.6: Acertando una pregunta

Figura B.7: Fallando una pregunta

será necesario superar la actividad dos veces con un balance de 6 puntos positivos.

Una vez pulsemos sobre el botón de la actividad del primer tipo, se nos hará una presentación con una muy breve descripción de la actividad. Si pulsamos start la partida dará comienzo y empezará el cronómetro a contar. Se nos mostrará una cuestión sobre un lugar que debemos identificar en las cuatro imágenes que tenemos en pantalla. Una vez localizada la que creamos que sea la imagen adecuada debemos pulsar su botón inmediatamente inferior.

Si hemos acertado se nos dará un punto y se mostrará la imagen (fig. B.6) en caso contrario se mostrará la imagen (fig. B.7) y tendremos un fallo en esta actividad. Debemos repetir este proceso hasta superar ocho preguntas o bien acabar por falta de tiempo, para que se nos muestre el resultado de la actividad. Una vez terminada una actividad podemos repetirla o salir al menú principal.

Para la actividad del segundo tipo, el comportamiento es idéntico salvo que esta vez deberemos reconocer qué foto es la que falta, y para ello debemos pulsar sobre el botón con el nombre que creamos que es el que falta. La respuesta del sistema será idéntica y mostrará el botón en verde

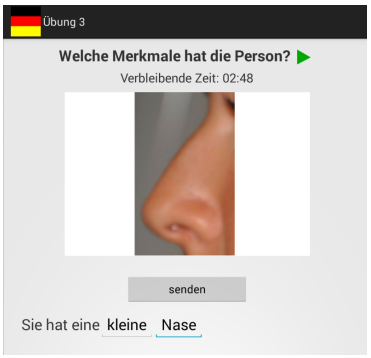


Figura B.8: Rellenando una respuesta

y nos asignará un punto en caso de acierto, y en caso de fallo mostrará el botón en rojo y nos anotará un fallo en la puntuación.

Para el tercer tipo de actividad debemos rellenar los huecos que faltan como se puede apreciar en la (fig. B.8) con el contenido que veamos que completa a la frase que aparece debajo de la imagen. Una vez rellenada la frase, debemos pulsar el botón *senden* para enviar nuestra propuesta al sistema para que la corrija. Si hemos acertado pasaremos a la siguiente pregunta. en el caso contrario, nos dirá que nos hemos equivocado y que repitamos de nuevo, aunque si hemos fallado tres veces nos proporcionará una solución para que podamos continuar aunque ya hayamos suspendido este intento.

Una vez terminado de jugar a las actividades, podemos consultar nuestro progreso en el apartado Punktezahl del menú principal del sistema. En éste apartado podemos ver nuestras puntuaciones y habrá que tener especial atención a los iconos que aparecen al lado de cada registro, si aparece un stick verde significará que tenemos esa actividad aprobada, y si aparece un cruz roja significará que está suspensa la actividad y debemos reintentarla.

B.5. Uso del chat

Una vez desbloqueado el chat, podremos jugar a la actividad *Catch me if you can*, para acceder a ella, debemos pulsar el botón Chat que aparece en el menú principal. Una vez pulsado, se nos mostrará una ventana de inicio de sesión (fig. B.9) como en el de las redes sociales. Debemos de escribir nuestro usuario y contraseña proporcionado por la profesora y pulsar el botón *login* para poder iniciar sesión.

Si todo ha ido correctamente y no ha habido problemas de conexión, nos saldrá la opción de empezar la partida o de salir del chat. Para jugar una partida debemos pulsar la opción empezar partida. Una vez dentro de la sala, se nos mostrará una vista similar a la de (fig. B.11). Dependiendo de en que orden hayamos entrado a la sala tendremos un rol u otro en el juego. Para

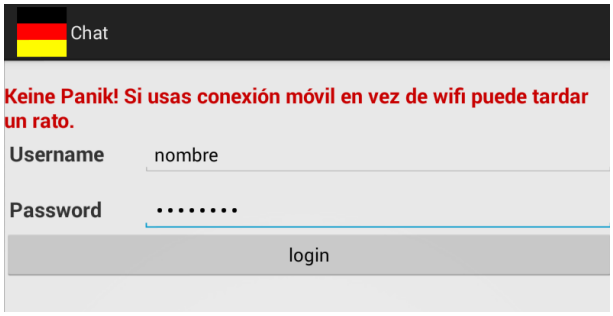


Figura B.9: Inicio de sesión

Figura B.10: Conversación en sala de chat

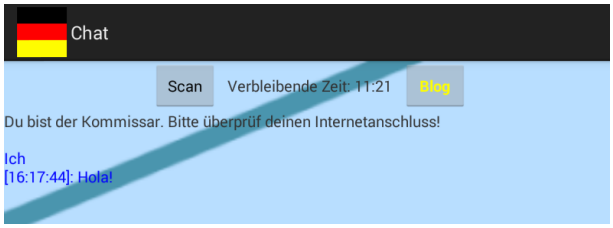


Figura B.11: Conversación en sala de chat

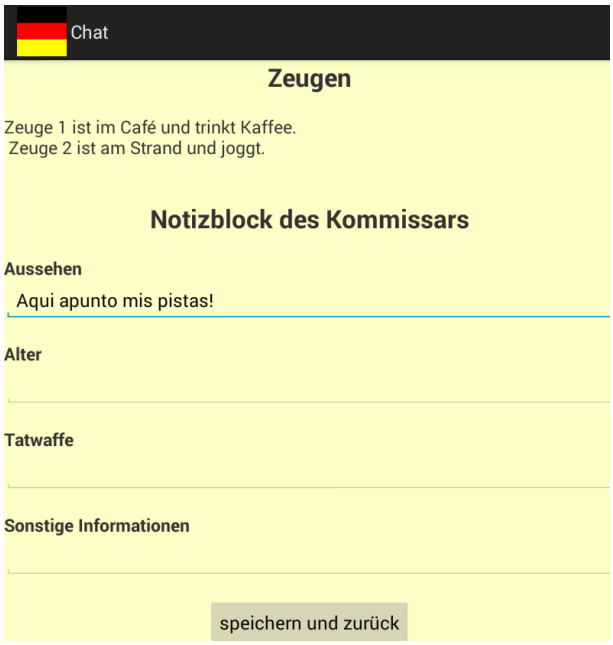


Figura B.12: Blog de notas del comisario

enviar un mensaje a la sala, debemos de escribir nuestro mensaje y pulsar el botón *senden*. Si todo ha ido correctamente los compañeros de la sala deberían haber recibido el mensaje.

Si nos ha tocado el rol del Comisario, tendremos la posibilidad de manejar el bloc de notas del comisario. En esta vista podremos apuntar nuestras pistas obtenidas de nuestros compañeros escribiéndolas tal y como viene en la (fig. B.12). Además, obtendremos los indicios con los que empezar a jugar a la actividad justo arriba, dónde pone Zeugen. Una vez terminado podemos pulsar el botón atrás de Android o sobre el botón *speichern und zurück*, ya que nuestras pistas se guardarán automáticamente.

Si deseamos escanear un código por que tengamos una pista delante, o queramos elegir entre uno de los sospechosos disponibles con códigos, sólo tenemos que pulsar en el botón Scan que aparece en la (fig. B.11) y se nos abrirá la aplicación para escanear códigos directamente. Una vez escaneado el código accederemos a una pista en forma de vídeo que podremos volver a reproducir pulsando el botón verde de play.

B.6. Desinstalación

Para la desinstalación de la aplicación no hace falta ningún requisito extra frente a otras aplicaciones descargadas desde Google play. Debemos irnos a nuestro apartado Ajustes dentro de nuestras aplicaciones Android. Una vez dentro del apartado ajustes de Android, debemos buscar el botón Aplicaciones, para poder elegir la aplicación a desinstalar. Buscamos la aplicación

VocabTrainer A1 y pulsamos sobre ella. Se nos abrirá un menú con diferentes opciones sobre la aplicación. Debemos pulsar sobre el botón Desinstalar y aceptaremos el diálogo que se nos muestre.

Una vez hecho este proceso la aplicación estará desinstalada de nuestro sistema. Si no vamos a necesitar más la aplicación Barcode Scanner por cualquier motivo, debemos repetir el proceso anterior pero buscando la aplicación Barcode Scanner, y se desinstalará de igual manera que VocabTrainer A1.

Apéndice C

Manual del profesor

En esta sección vamos a hablar de los pasos necesarios para gestionar la aplicación y los juegos por parte del profesor.

C.1. Creación de códigos QR

Para crear un código QR, podemos utilizar una de las muchas aplicaciones disponibles para ello, bien vía web o escritorio. Yo he utilizado para agilizar su uso la web `http://qrcode.kaywa.com`, en esta web se nos presenta un recuadro en blanco y un marco de texto dónde introducir nuestro código.

Como vemos en la foto de ejemplo (Fig. C.1), si queremos crear el código 2001, simplemente debemos escribir 2001, pero es muy importante pulsar el botón static, ya que así no nos pedirá que nos registremos para poder modificar nuestros códigos, nosotros sólo vamos a crear uno de ejemplo. Cuando hayamos escrito el código pulsamos *generate* y aparecerá nuestro código.

Una vez haya recargado la página, veremos nuestro código en dónde había antes un cuadrado en

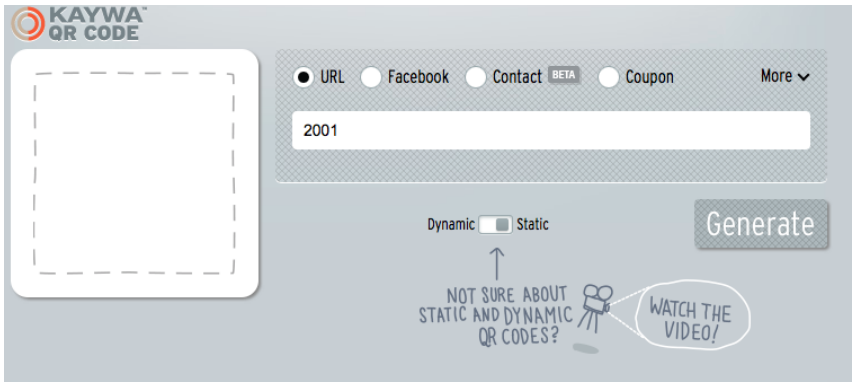


Figura C.1: Página para la creación de códigos QR

blanco. Ahora, para guardarlo, simplemente pulsamos el click derecho del ratón sobre el código y lo guardamos a gusto del usuario.

C.2. Añadir temario: texto

Para añadir más temario, tendremos que modificar también los ficheros de recursos. En el directorio 'values' tendremos los ficheros fills, hidden y search que representan el contenido de cada una de las actividades que tiene la aplicación. En cada fichero, hay definido un recurso que define el número de preguntas posibles que tiene esa actividad. Es decir que si el recurso 'search_maxplaces' tiene definido como valor el número 22, significa que hay 22 preguntas disponibles para hacerle al usuario.

Si quisiéramos añadir una más, tendríamos que cambiar el número 22 por el número 23, y evidentemente añadir una pregunta más. Hay que remarcar que las preguntas pertenecientes al primer temario van del rango **1-100**, las del segundo temario **101-200** y las del tercer temario al rango **201-300**. Cada pregunta está definida bajo una estructura de recursos, en el caso de la actividad search, es de la forma siguiente:

```
1 <string name=''search_question22''>einen Hof</string>
```

Ese recurso representa la última pregunta que tenemos en el apartado search para el primer temario de la aplicación. Por lo tanto, si queremos añadir una nueva, que fuese 'la playa' sería de la siguiente manera:

```
1 <string name=''search_question23''>la playa</string>
```

De esta manera, la aplicación entiende que tiene disponibles 23 preguntas y sabe cual es el texto de la pregunta número 23. Expongo ejemplos para actividades del tipo Fill y Hidden:

```
1  <string name=''hidden_question22''>der Hof</string>
```

Para añadir una más:

```
1  <string name=''hidden_question23''>la playa</string>
```

Y por último expongo un breve ejemplo de la del tipo Fill:

```
1  <string name=''fill_question24.1''>Er </string>
   <string name=''fill_question24.1''> ein Eis im </string>
3  <string name=''fill_firstpart24''>isst</string>
   <string name=''fill_secondpart24''>Hof</string>
```

Se puede apreciar que tiene una primera parte, y una segunda parte a rellenar por el usuario, por lo que añadir una nueva sería algo parecido a esto:

```
   <string name=''fill_question25.1''>Ella </string>
2  <string name=''fill_question25.1''> por el </string>
   <string name=''fill_firstpart25''>pasea</string>
4  <string name=''fill_secondpart5''>parque</string>
```

C.3. Añadir temario: sonido e imágenes

En el subapartado anterior comentábamos la posibilidad de añadir temario metiendo más recursos de texto. Cada pregunta nueva que añadamos a los recursos, debe venir acompañada de su respectiva imagen y sonido dependiendo del tipo de ejercicio.

Los sonidos, que se ubican en el directorio raw, tienen un formato específico. Si por ejemplo nos fijamos en el archivo 'search_audio22.mp3' significa que ese fichero contiene el audio que describe al objeto 22 del apartado search. Si hemos añadido una pregunta número 23 al apartado search, debemos de aportar el sonido que describe a ese nuevo objeto que queramos añadir. Nos quedaría un fichero llamado 'search_audio23.mp3' que será completamente nuevo para la aplicación.

Esto es aplicable al resto de ficheros, como los 'search_question...' que contiene la pregunta que se le realiza al alumno conforme al objeto que le toca responder. Por lo tanto, si hemos añadido el número 23 debemos aportar nuestro 'search_audio23.mp3' y 'search_question_audio23'.

Con las imágenes pasa exactamente lo mismo, si hemos añadido un objeto para el apartado search número 23, debemos añadir una imagen a la carpeta de recursos de imágenes (en nuestro caso drawable-hdpi) con el nombre search_image23.jpg debidamente redimensionada y comprimida a jpg.

Con estos pasos descritos podríamos añadir más temario a la aplicación o simplemente modificar el que ya tenemos, por lo que le da una gran versatilidad a la aplicación.

C.4. Añadir actividades a Catch me, if you can!

Ya hemos comentado como llegamos a implementar la definición de actividades para *Catch me, if you can!*, ahora vamos a comentar como añadir más actividades al juego colaborativo.

Primero tenemos que acceder al fichero assignment.xml de la carpeta res/values del proyecto de Android. Una vez localizado tendremos que modificar y añadir recursos para implementar una nueva actividad. Comencemos con el máximo número de actividades que nos podemos encontrar:

```
<integer name="max">2</integer>
```

Como vemos, como máximo sólo habrá dos actividades que el sistema utilizará, por lo tanto si queremos añadir una nueva actividad debemos modificar el número 2 por el número 3, 4, o por el número de actividades que vaya a implementar el juego. Una vez hecho eso debemos añadir los indicios propios de esa actividad.

Pongamos como ejemplo que nuestra nueva actividad es la número 3 (para seguir con el orden), tendríamos que definir un número de parejas de indicios (ya que hay dos policías y colaboran

entre ellos con los indicios que les proporciona el comisario).

```
    <string name="assignment3.0">
        Zeuge 1 Texto del indicio 1. "\n"
        Zeuge 2 Texto del indicio 2. "\n"
    </string>

    <string name="assignment3.1">
        Zeuge 3 Texto del indicio 3 "\n"
        Zeuge 4 Texto del indicio 4. "\n"
    </string>

    <string name="assignment3.2">
        Zeuge 5 Texto del indicio 5 "\n"
        Zeuge 6 Texto del indicio 6 "\n"
    </string>

    <string name="assignment3.3">
        Zeuge 7 Texto del indicio 7 "\n"
        Zeuge 8 Texto del indicio 8 "\n"
    </string>

    <string name="assignment3.4">
        Zeuge 9 Texto del indicio 9 "\n"
        Zeuge 10 Texto del indicio 10 "\n"
    </string>

    <string name="assignment3.5">
        Zeuge 11 Texto del indicio 11 "\n"
        Zeuge 12 Texto del indicio 12 "\n"
    </string>
```

Cada par, viene definido por la cadena de texto ‘assignmentx.y’ . La x, en nuestro caso el número 3, define la actividad a la que pertenece nuestro grupo de indicios. El segundo número indica la pareja de indicio que es, empezando por el número 0. Como ejemplo hemos definido seis pares, doce indicios. También tendremos que definir ese dato.

```
<integer name="amount3">12</integer>
```

Una vez definido los textos de los indicios que verá el comisario, tenemos que definir la confirmación que aparecerá en la sala cuando un policía escanee uno de esos indicios. Expongamos un ejemplo:

```
    <string name="assignmentanswer1.1">Se ha encontrado la pista numero 1 </string>
    <string name="assignmentanswer1.2">Se ha encontrado la pista numero 2 </string>
    <string name="assignmentanswer1.3">Se ha encontrado la pista numero 3 </string>
    <string name="assignmentanswer1.4">Se ha encontrado la pista numero 4 </string>
    <string name="assignmentanswer1.5">Se ha encontrado la pista numero 5 </string>
    <string name="assignmentanswer1.6">Se ha encontrado la pista numero 6 </string>
    <string name="assignmentanswer1.7">Se ha encontrado la pista numero 7 </string>
    <string name="assignmentanswer1.8">Se ha encontrado la pista numero 8 </string>
    <string name="assignmentanswer1.9">Se ha encontrado la pista numero 9 </string>
    <string name="assignmentanswer1.10">Se ha encontrado la pista numero 10 </string>
    <string name="assignmentanswer1.11">Se ha encontrado la pista numero 11 </string>
    <string name="assignmentanswer1.12">Se ha encontrado la pista numero 12 </string>
```

En este caso, hemos definido simplemente un texto de confirmación genérico para cada pista encontrada, alertando de que se ha escaneado una pista.

A cada pista escaneada, el policía podrá visualizar un vídeo, un vídeo prefijado que contendrá la pista con la información relevante. Debemos definir el nombre que tiene el vídeo y que también contendrá el código, para que se produzca la reproducción de x vídeo al escanear x código.

```
<string name="assignmentcode3.1">1001</string>
<string name="assignmentcode3.2">1002</string>
<string name="assignmentcode3.3">1003</string>
<string name="assignmentcode3.4">1004</string>
<string name="assignmentcode3.5">1005</string>
<string name="assignmentcode3.6">1006</string>
<string name="assignmentcode3.7">1007</string>
<string name="assignmentcode3.8">1008</string>
<string name="assignmentcode3.9">1009</string>
<string name="assignmentcode3.10">1010</string>
<string name="assignmentcode3.11">1011</string>
<string name="assignmentcode3.12">1012</string>
<string name="assignmentcode3.20">1020</string>
```

También tendremos que añadir los nodos que representan los estados por los que la actividad pasará mientras es jugada. La secuencia que definamos en el fichero assignment.xml será la secuencia que seguirán los vídeos/pistas. Como ejemplo ponemos:

```
<string name="adjacency3.1">1,2,3,4</string>
<string name="adjacency3.2">1,2,3,4</string>
<string name="adjacency3.3">3,4,5,6</string>
<string name="adjacency3.4">3,4,5,6</string>
<string name="adjacency3.5">5,6,7,8</string>
<string name="adjacency3.6">5,6,7,8</string>
<string name="adjacency3.7">7,8,9,10</string>
<string name="adjacency3.8">7,8,9,10</string>
<string name="adjacency3.9">9,10,11,12</string>
<string name="adjacency3.10">9,10,11,12</string>
<string name="adjacency3.11">11,12</string>
<string name="adjacency3.12">11,12</string>
```

Como ejemplo hemos puesto que los nodos sean completamente secuenciales, tal y como ocurre con las otras dos actividades, pero podríamos cambiar el orden si quisiéramos e incluso crear ciclos que harían que la actividad nunca terminase (aunque no vaya a tener ninguna aplicación real).

Ya por último, debemos indicar que nodos van a ser los iniciales:

```
<string name="initialnodes3">1,2,3,4</string>
```

En este ejemplo, podremos empezar por los cuatro primeros nodos (por que lo hemos elegido así para el ejemplo) aunque bien podría variar al gusto del que desarrolle la actividad.

Una vez seguido estos pasos, ya estaría definida la actividad, aunque sería necesario añadir los vídeos representados por los códigos definidos por 'assignmentcode3.x' ya que son el contenido que van a obtener los policías al escanear los códigos.

[16:29] Kommissar (usuario2) ha ingresado al cuarto
[16:29] PolizistdXN1YXJpbzE= (usuario1) ha ingresado al cuarto
[16:29] PolizistdXN1YXJpbzM= (usuario3) ha ingresado al cuarto
[16:29] Kommissar: hallo!
[16:29] PolizistdXN1YXJpbzM=: hallo¡
[16:30] PolizistdXN1YXJpbzE=: hallo
[16:30] Kommissar: Zeuge 1 für Polizist 1: es ist im Cafe und trinkt Kaffee
[16:31] PolizistdXN1YXJpbzM=: gehe ich
[16:31] Kommissar: Zeuge 2 ist am Strand und joggt
[16:31] PolizistdXN1YXJpbzE=:
c0de_X3Rlc3QyX3VzdWFyaW8xQHZwczlzNjctY2xvdWQuY29tYWxpcy5uZXRfMTAwMQ==

[16:31] Kommissar: Ja, du hast Zeuge 1 gefunden
[16:31] PolizistdXN1YXJpbzM=:
c0de_X3Rlc3QyX3VzdWFyaW8zQHZwczlzNjctY2xvdWQuY29tYWxpcy5uZXRfMTAwMg==

Figura C.2: Comienzo de una conversación

C.5. Registros de conversaciones

Ya hemos comentado que el servidor almacena los registros de las conversaciones, y hemos visto cómo acceder a ellos, pero no hemos mostrado ningún ejemplo de ellos. El formato en el que podemos almacenarlos en nuestro equipo es en formato PDF, y podremos ver todas las entrañas de la conversación.

Cómo se puede ver en la (Fig. C.2), podemos ver como entran al servidor de chat tres usuarios. El primero adquirirá el rol de Comisario, que en este caso es el usuario registrado en el servidor llamado: usuario2. Una vez que se unen los otros dos jugadores, se les denomina como polizist y una cadena que simboliza su nombre codificado, para evitar que puedan conocerse entre ellos.

Podemos ver que se saludan y el Comisario comienza a mandar indicaciones a los policías. Estos, escanean los códigos que encuentran repartidos por el terreno de juego, y obtienen su confirmación con el mensaje *Ja, du hast Zeuge 1 gefunden*. Los policías, envían la información obtenida de las pistas en formato audio, de manera escrita como se puede ver en la (Fig. C.3). De esta manera el comisario anota la información obtenida y sigue enviando más información a los policías.

Por último, y cuando el comisario tiene suficientes pistas, decide escanear al posible asesino, para obtener respuesta positiva con: *Der Kommissar hat den Mörder identifiziert.*. De esta manera

[16:34] Kommissar: Nein, das ist nicht Zeuge
[16:34] PolizistdXN1YXJpbzE=: die Person hat smalles Gesicht
[16:35] PolizistdXN1YXJpbzE=: und ein tatowierung
[16:35] Kommissar: in der Kneipe und rauch
[16:35] PolizistdXN1YXJpbzM=:
c0de_X3Rlc3QyX3VzdWFyaW8zQHZwczIzNjctY2xvdWQuY29tYWxpcy5uZXRfMTAwNQ==

Figura C.3: Transcripción de una conversación

[16:38] Kommissar: Ja, du hast Zeuge 8 gefunden
[16:39] PolizistdXN1YXJpbzE=: die Person hat drei Tage Barbe
**[16:39] Kommissar: Der Kommissar hat den Mrder identifiziert.
Notizblock des Kommissars:**
**
Notiz 0: die Person hat kurzes Haar die Person hat eun schales Gesicht kleine Nase klein Mund smallest Gesicht ein tatowierung braunes Haar braune augenbrauen rote augen die person hört normalerweise musik</br>
Notiz 1: </br>
Notiz 2: </br>
Notiz 3: </br></br>**
[16:39] PolizistdXN1YXJpbzM=: super¡¡¡
[16:40] PolizistdXN1YXJpbzM= (usuario3) se ha ido del cuarto
[16:53] Kommissar (usuario2) se ha ido del cuarto
[16:53] PolizistdXN1YXJpbzE= (usuario1) se ha ido del cuarto

Figura C.4: Final de una conversación

el juego confirma que se ha encontrado el asesino y la partida ha terminado. Para dar la información de las notas apuntadas por parte del Comisario al profesor que supervise la actividad, se muestra por pantalla el blog completo a partir de la secuencia *Notizblock des Kommissars:*.

De esta manera se puede almacenar diferentes conversaciones separadas por actividades y poderse analizar la participación real de cada jugador, evitando trampas o dejando que toda la carga del juego la lleve un sólo jugador.

Apéndice D

Difusión

D.1. Divulgación

El proyecto ha tenido repercusión mediática en la provincia de Cádiz debido a las pruebas que hemos ido realizando con los diferentes alumnos de diferentes instituciones. Las publicaciones que el proyecto ha obtenido son las siguientes:

- Publicación en el diario de Cádiz: http://www.diariodecadiz.es/article/cadiz/1734457/estudiantes/la/uca/aprenden/idiomas/jugando/con/una/app/para/moviles.html
- Publicación en el diario de Jerez: http://www.diariodejerez.es/article/provincia/1734883/estudiantes/la/uca/aprenden/idiomas/jugando/con/una/app/para/moviles.html
- Publicación en el diario de Cádiz, sección Chiclana: http://www.diariodecadiz.es/article/chiclana/1738984/una/nueva/actividad/ies/poeta/para/aprendizaje/idiomas.html
- Noticia en portada de la web de la UCA: http://www.uca.es/es/cargarAplicacionNoticia.do?identificador=6578

D.2. Congresos científicos

Internacionalmente, el proyecto ha sido aprobado para dos publicaciones europeas, siendo éstas:

- EUROCALL 2014, CALL Design: Principles and Practice, 20.-23 august, University of Groningen https://www.eurocall2014.nl
- Antwerp CALL 2014, Research Challenges in Call. 7.-9. july, University of Antwerpen. http://www.ua.ac.be/main.aspx?c=.CALL2010

Bibliografía

[1] Gargenta, M. & Nakamura, M. (2014). Learning Android, 2nd Edition. Editorial O'Reilly Media. (ISBN: 978-1449319236)

[2] Naftalim,M. & Walder, P. (2006). Java Generics and Collections Editorial O'Reilly Vlg Gmbh & Co (ISBN: 978-0596527754)

[3] Sharma, M. (2008). Openfire administration. Editorial: Packt Publishing. (ISBN: 978-1847195265)

[4] Saint-Andre, P., Smith, K. & Tronçon, R. (2009). XMPP: The Definitive Guide, Editorial O'Reilly. (ISBN: 978-0596521264)

[5] Juho Hamari, J. (2014). Investigación sobre la gamificación y sus resultados. 47th Hawaii International Conference on System Sciences 2014 (retreieved in: http://gamification-research.org/2013/09/does-gamification-work-a-look-into-research/)

[6] Tassos A. Mikropoulos (2009), Estudio de diez años sobre los mundos virtuales en la educación, Computers & Education, 2009 http://www.sciencedirect.com/science/article/pii/S0360131510003052

[7] Andrzej Marczewski, Comparativa entre los videojuegos y la gamificación, Gamasutra, 2013 http://www.gamasutra.com/blogs/AndrzejMarczewski/20131024/203054/Game_Thinking__Breaking_down_gamification_and_games.php

[8] Javadoc de la documentación de Smack http://www.igniterealtime.org/builds/smack/docs/latest/javadoc/

[9] Primeros pasos a la hora de desarrollar en Android http://developer.android.com/training/index.html

[10] Whitton, N. (2010). Learning with digital games: A practical guide to engaging students in Higher Education, London: Routledge. (ISBN 9780415997751)

[11] Holden, C., & Sykes, J (2011). Leveraging mobile games for place-based language learning, International Journal of Game-Based Learning, 1(2), 1–18

[12] Stockwell, G. (2010). Using mobile phones for vocabulary activities: Examining the effect of the platform. Language Learning & Technology, 14 (2), 95–110

[13] Levy, M., & Kennedy, C. (2005) Learning Italian via mobile SMS. In A. Kukulska-Hulme & J. Traxler (Eds.), Mobile learning: A handbook for educators and trainers (pp. 76–83). London: Routledge

[14] Thornton, P., & Houser, C. (2005) Using mobile phones in English education in Japan. Journal of Computer Assisted Learning, 21(3), 217–228

[15] Kukulska-Hulme, A. (2005) Mobile usability and user experience. In A. Kukulska-Hulme & J.Traxler (Eds.), Mobile learning: A handbook for educators and trainers (pp. 45–56). London: Routledge

[16] Berns, A., Palomo-Duarte, Dodero, J.M., Valero-Franco, C. (2013) Using a 3D Online Game to Assess Students' Foreign Language Acquisition and Communicative Competence. In Hernández-Leo et al. (Eds.): EC-TEL 2013, LNCS 8095, pp. 19–31, Springer-Verlag Berlin Heidelberg 2013 (D.O.I. 10.1007/978-3-642-40814-4_3)

[17] Cornillie, F., Thorne, S. L., & Desmet, P. (2012) Digital games for language learning: from hype to insight? ReCALL 24 (3), 243-256 (doi:10.1017/S0958344012000134)

[18] Pellerin, M. (2013) Using mobile technologies to promote authentic oral language learning and new forms of language assessment. Proceedings of WorldCALL 2013. Global perspectives on Computer-Assisted Language Learning, Glasgow, 10-13 July 2013, 276-279

[19] Reinders, H. & Wattana, S. (2011) 'Learn English or Die: The effects of digital games on Interaction and Willingness to Communicate in a Foreign Language'. Digital Culture and Education, 3(1), p. 4-28

[20] Cornillie, F., Clarebout, G., Desmet, P. (2012) The role of feedback in foreign language learning through digital role playing games. Procedia Social and Behavioral Sciences, (34) 49–53

[21] Burston, J. Twenty years of MALL project implementation: A meta-analysis of learning outcomes. ReCALL / FirstView Article pp 1-17

[22] Gonzalez-Barahona, J.M., Fernandez-Gonzalez, J., Robles, G. (2011) Implementing Gymkhanas with Android Smartphones: a Multimedia M-Learning Game. IEEE Engineering Education (EDUCON) (ISBN: 978-1-61284-642-2)

GNU Free Documentation License

Version 1.3, 3 November 2008

Preamble

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.
This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.
We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "**Document**", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "**you**". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.
A "**Modified Version**" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "**Secondary Section**" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "**Invariant Sections**" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "**Cover Texts**" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "**Transparent**" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "**Opaque**".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "**Title Page**" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "**publisher**" means any person or entity that distributes copies of the Document to the public.

A section "**Entitled XYZ**" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "**Acknowledgements**", "**Dedications**", "**Endorsements**", or "**History**".) To "**Preserve the Title**" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.
You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.
If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.
If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.
It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.

C. State on the Title page the name of the publisher of the Modified Version, as the publisher.

D. Preserve all the copyright notices of the Document.

E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.

F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.

G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.

H. Include an unaltered copy of this License.

I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.

J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements”.

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.
You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.
If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.
If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.
Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.
Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See `http://www.gnu.org/copyleft/`.
Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.
"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.
"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.
An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright © YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with ... Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.
If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.